Vysoká škola báňská - Technická univerzita Ostrava

Fakulta Strojní

Katedra automatizační techniky a řízení

Softwarové moduly obsluhy vybraných periférií pro mikroprocesory STM 32

Programming Support of STM32F4 Microprocessor Peripherals

Student: **Václav Chrascina**

Vedoucí bakalářské práce: **Ing. David Fojtík, Ph.D.**

Ostrava 2015

VŠB - Technická univerzita Ostrava
Fakulta strojní
Katedra automatizační techniky a řízení

Zadání bakalářské práce

Student: **Václav Chrascina**

Studijní program: B2341 Strojírenství

Studijní obor: 3902R001 Aplikovaná informatika a řízení

Téma: Programová podpora periferií mikroprocesorů řady STM32F4
Programming Support of STM32F4 Microprocessors Peripherals

Jazyk vypracování: čeština

Zásady pro vypracování:

1. Seznamte se s vývojovým prostředím KEIL a podpůrnou knihovnou STM32Cube se zaměřením na programování mikroprocesorů řady STM32F4.
2. Vytvořte přehled dostupných periferií a vývojových kitů procesorů řady STM32F4 a pro vybrané z nich zpracujte podrobný popis.
3. Pro vybrané periferie a vývojové kity s využitím knihovny STM32Cube vytvořte sadu jednoduchých demonstrujících úloh.
4. Zpracujte studijní opory vytvořených příkladů a zhodnoťte dosažené výsledky.

Seznam doporučené odborné literatury:

ARM KEIL. *Getting Started Create Applications with MDK Version 5 for ARM® Cortex®-M Microcontrollers.* ARM Germany GmbH, 2015 dostupné z <URL: http://www2.keil.com/docs/default-source/default-document-library/mdk5-getting-started.pdf>
HEROUT, P. *Učebnice jazyka C.* České Budějovice, nakladatelství KOPP, září 2004, IV. přepracované vydání, ISBN 80-7232-220-6, 280 stran.
SEAL, DAVID. *ARM architecture reference manual.* 2nd ed. Harlow: Addison-Wesley, 2001, 1 sv. (různé stránkování). ISBN 0-201-73719-1.
STMICROELECTRONICS *Microcontrollers*, technická dokumentace dostupné z <URL: http://www.st.com/web/en/catalog/mmc/FM141>
STMICROELECTRONICS *STM32CubeF4*, technická dokumentace dostupné z <URL: http://www.st.com/web/catalog/tools/FM147/CL1794/SC961/SS1743/LN1897/PF259243>
VÁŇA, VLADIMÍR. ARM pro začátečníky. 1. vyd. Praha: BEN - technická literatura, 2009, 195 s. ISBN 978-80-7300-246-6.

Formální náležitosti a rozsah bakalářské práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí bakalářské práce: **Ing. David Fojtík, Ph.D.**

Datum zadání: 11.12.2015

Datum odevzdání: 16.05.2016

Renata Wagnerová

doc. Ing. Renata Wagnerová, Ph.D.
vedoucí katedry

VYSOKÁ ŠKOLA BÁŇSKÁ-TECHNICKÁ UNIVERZITA Fakulta strojní 3 OSTRAVA

doc. Ing. Ivo Hlavatý, Ph.D.
děkan fakulty

Prohlášení diplomanta

Prohlašuji, že jsem celou diplomovou (bakalářskou) práci včetně příloh vypracoval samostatně pod vedením vedoucího diplomové (bakalářské) práce a uvedl jsem všechny použité podklady a literaturu.

V Ostravě…… ……

Václav Chrascina

Prohlašuji, že

- byl jsem seznámen s tím, že na moji diplomovou (bakalářskou) práci se plně vztahuje zákon č. 121/2000 Sb. – autorský zákon, zejména §35 – užití díla v rámci občanských a náboženských obřadů, v rámci školních představení a užití díla školního a §60 – školní dílo.
- beru na vědomí, že Vysoká škola báňská – Technická univerzita Ostrava (dále jen VŠB-TUO) má právo nevýdělečně ke své vnitřní potřebě diplomovou (bakalářskou) práci užít (§35 odst. 3).
- souhlasím s tím, že jeden výtisk diplomové (bakalářské) práce bude uložen v Ústřední knihovně VŠB-TUO k prezenčnímu nahlédnutí a jeden výtisk bude uložen u vedoucího diplomové (bakalářské) práce. Souhlasím s tím, že údaje o diplomové (bakalářské) práci, obsažené v Záznamu o závěrečné práci, umístěném v příloze mé diplomové (bakalářské) práce, budou zveřejněny v informačním systému VŠB-TUO.
- bylo sjednáno, že s VŠB-TUO, v případě zájmu z její strany, uzavřu licenční smlouvu s oprávněním užít dílo v rozsahu §12 odst. 4 autorského zákona.
- bylo sjednáno, že užít své dílo – diplomovou (bakalářskou) práci nebo poskytnout licenci k jejímu využití mohu jen se souhlasem VŠB-TUO, která je oprávněna v takovém případě ode mne požadovat přiměřený příspěvek na úhradu nákladů, které byly VŠB-TUO na vytvoření díla vynaloženy (až do jejich skutečné výše).
- beru na vědomí, že odevzdáním své práce souhlasím se zveřejněním své práce podle zákona č. 111/1998 Sb., o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších předpisů, bez ohledu na výsledek její obhajoby.

V Ostravě ……. ……..……

Václav Chrascina

Václav Chrascina

Písek u Jablunkova č. p. 310

739 84

ANOTOACE BAKALÁŘSKÉ PRÁCE

CHRASCINA, V. *Programová podpora periférií mikroprocesorů řady STM32F4.* Ostrava: katedra automatizační techniky a řízení, Fakulta strojní VŠB – Technická univerzita Ostrava, 2016, 42 s. Bakalářská práce, vedoucí Ing. David Fojtík, Ph.D.

Bakalářská práce se zabývá problematikou programování vybraných periférií mikroprocesorů řady STM32F4 s využitím STM32Cube Embeded Software. Hlavní náplní této práce je vytvoření sady demonstračních úloh, které byly zpracovány do podoby studijních materiálů. Práce se obzvláště věnuje ovládáním standartního vstupu a výstupu a ovládáním LCD dotykového displeje. Úlohy byly vytvořeny pro desku STM32F4 – Discovery pomocí programu KEIL µVision 5.

Teoretická část práce se věnuje popisu na trhu dostupných procesorů řady STM32F4. Je zde vysvětleno k jakému účelu se nejvíce vhodné, jaké používají jádro a periférie, a zda jsou kompatibilní s ostatními produkty. Teoretická část práce dále obsahuje i seznam a popis vybraných periférií mikroprocesorů řady STM32F4. V teoretické části je zpracován také popis vývojového prostředí Keil µVision 5 a podpůrné knihovny STM32Cube.

Klíčová slova: programování, STM32F4, STM32Cube, KEIL, STemWin

ANNOTATION OF THE BACHELOR WORK

CHRASCINA, V. *Programming Support of STM32F4 Microprocessor Peripherals.* Depatrment of Control Systems and Instrumentation, Faculty of Mechanical Engineering VŠB – Technical University of Ostrava 42 p. Bachelor thesis, head: Ing. David Fojtík, Ph.D.

This thesis deals with problems of programming selected peripherals of STM32F4 microprocessors using STM32Cube Embeded Software. Main goal of the thesis is creating a set of demonstration exercises, which was transformed into studying materials. Thesis is especially focused on controlling of general input and output and controlling of LCD touchscreen. Exercises were created for STM32F4 – Discovery kit using Keil µVision 5 environment.

Theoretical part of thesis is focused on description of STM32F4 microprocessors. It explains which purposes they are suitable for, which core and peripherals are they using, and if are they compatible with other products. Theoretical part also includes list and description of selected STM32F4 processor peripherals. Theoretical part also includes description of Keil µVision 5 environment and STM32Cube library.

Keywords: programming, STM32F4, STM32Cube, KEIL, STemWin

Obsah bakalářské práce

Seznam použitých zkratek

ADC	(Analog-to-Digital Converter) Analogově digitální převodník
ART	(Adaptive Real-Time).
CAN	(Controller Area Network) Průmyslová komunikační síť.
DAC	(Digital-to-Analog Converter) Digitálně analogový převodník.
DMIPS	(Dhrystone Millions of Instructions Per Second). Milióny operací za sekundu v benchmark programu Dhrystone
DSP	(Digital Signal Processor) Procesor určený ke zpracování digitálních signálů.
EEPROM	(Electrically Erasable Programmable Read-Only Memory) Paměť, jejíž obsah se uchová i po vypnutí napájení.
FPU	(Floating Point Unit) Koprocesor operující s čísly s pohyblivou řádovou čárkou.
MCU	(Mictrocontroler Unit) Mikrokontroler.
MEMS	(Micro-Electro-Mechanical Systems).
NVM	(NonVolatile Memory) Paměť jejíž obsah se uchovává i po přerušení napájení.
RAM	(Random-Access Memory) Paměť s přímým přístupem.
SRAM	(Static Random Access Memory) Paměť, jejíž obsah se po vypnutí napájení zapomene
STM	STMicroelectronics.
UART	(Universal Synchronous / Asynchronous Receiver and Transmitter) Synchronní / asynchronní sériové rozhraní.
USB	(Universal Serial Bus) Univerzální sériová sběrnice.

1 Úvod

Mikroprocesory jsou dnes již zastoupeny téměř v každém odvětví lidské činnosti především díky mikrokontrolerům. Nízká cena a nárust výpočetního výkonu pomohly k expanzi mikrokontrolerů, které dnes řídí výrobní linky, televize, rádia nebo telefony. Další výhodou těchto MCU je možnost připojení různých periférií, se kterými je možno komunikovat prostřednictvím různých způsobů. Mezi tyto periférie patří: LCD displej, USB port, sériová linka aj.

Cílem této práce je vytvořit sadu demonstrujících úloh pro vybrané periférie a vývojové kity a tyto úlohy přetvořit do návodů pro začátečníky. Pro splnění tohoto úkolu je nutné se nejdříve seznámit s vývojovým prostředím Keil a také s podpůrnou knihovnou STM32Cube. Teoretická část této práce si klade za cíl zpracovat popis periférií a vývojových kitů řady STM32.

2 Procesory řady STM32

STM32 je řada mikrokontrolerů vyráběná firmou STMicroelectronics. Tyto mikrokontrolery pracují s 32-bitovým jádrem, postaveným na architektuře ARM. Řada se dále dělí na řadu s nízkou spotřebou: STM32 L0, STM32 L1, STM32 L4, standartní řadu: STM32 F0, STM32 F1, STM32 F3 a na řadu s vysokým výkonem: STM32 F2, STM32 F4, STM32 F7.

2.1 Procesory řady s nízkou spotřebou

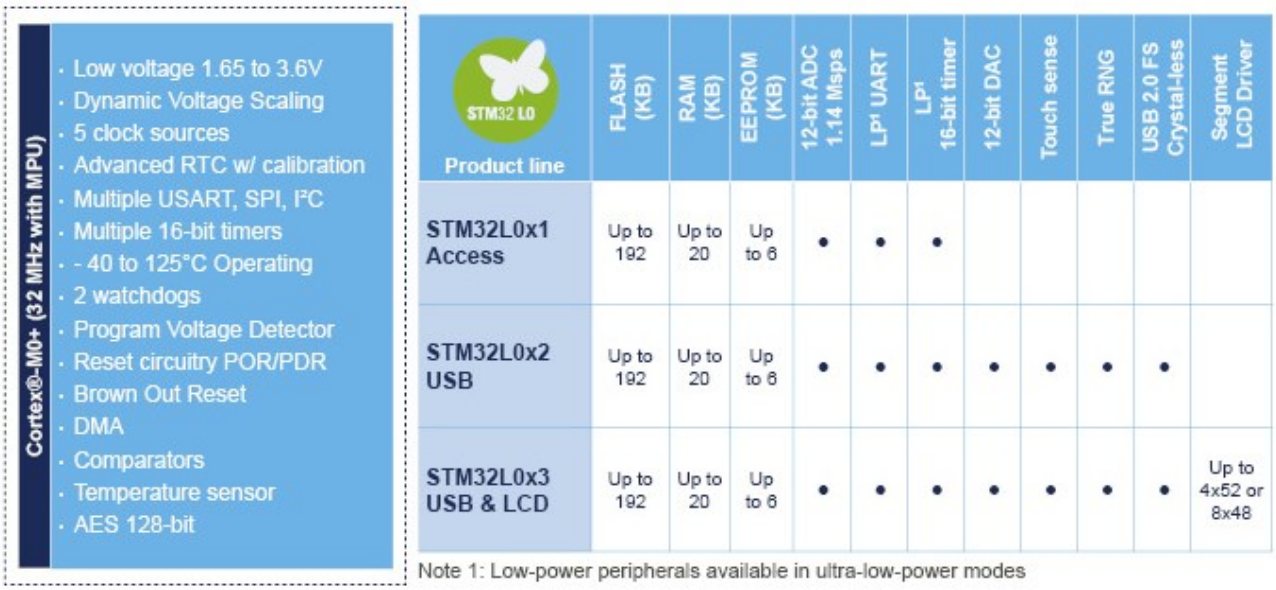
Mezi mikrokontrolery řady s nízkou spotřebou patří: STM32 L0, STM32 L1 a STM32 L4.

STM32 L0

Procesory série STM32 L0 jsou navrženy s důrazem na nízkou spotřebu baterie při zachování optimální výkonnosti. Kombinují jádro ARM® Cortex®-M0+ a komponenty s nízkou spotřebou energie, proto se nejlépe hodí do aplikací napájených baterií nebo aplikací, kde se energie získává z okolí (energy harvesting).

STM32 L0 nabízí dynamické škálování napětí, generátor hodin s nízkou spotřebou, LCD rozhraní, komparátor, DAC, hardwarové šifrování, teplotní sensor, 128b AES, DMA, 2 watchdogy, provozní teploty od -40°C do 125°C, 16b časovače, USART, I²C nebo dotykové rozhraní.

Tyto procesory jsou k dispozici s 64KB Flash paměti, 8KB RAM a 2KB vestavěné EEPROM.



Obr. 1 Řada MCU STM32 L0 [STMicroelectronics]

STM32 L1

Řada produktů STM32 L1 rozšiřuje koncept MCU s nízkou spotřebou a klade důraz na výkon. STM32 L1 používá jádro ARM® Cortex®-M3 a stejně jako STM32 L0 a STM8 L nabízí dynamické škálování napětí, generátor hodin s nízkou spotřebou, LCD rozhraní, komparátor, DAC a hardware šifrování.

Tato řada nabízí široké portfolio prvků, velikostí pamětí a pin balíčků. Portfolio zahrnuje 32 až 512 Kb Flash paměti (80 Kb SDRAM a 16 Kb vestavěné EEPROM) a od 48 do 144 pinů. Velké množství vestavěných periférií jako USB, LCD rozhraní, operační zesilovač, komparátory, ADC s rychlým ON/OFF módem, DAC, kapacitní dotyk a AES dovolují rozšíření platformy a pin-to-pin kompatibilita se sérií STM32 F nabízí možnost plnohodnotného STM32 systému.

Série je dostupná v těchto řadách: STM32L100, STM32L151, STM32L152 (LCD), STM32L162 (LCD and AES-128).

Cortex®-M3 (32 MHz with MPU)
- Low voltage 1.65 to 3.6V
- Dynamic Voltage Scaling
- 5 clock sources
- Advanced RTC w/ calibration
- Multiple USART, SPI, I²C
- 16- and 32-bit timers
- - 40 to 85°C Operating Up to 105°C in LP Modes
- 2 watchdogs
- Brown Out Reset
- Program Voltage Detector (PVD)
- DMA
- Reset circuitry POR/PDR
- 12-bit ADC 1 MSPS
- 12-bit DAC

Product line	FLASH (KB)	RAM (KB)	EEPROM (KB)	Memory I/F	Op-Amp	Comp.	Temp. Sensor	Capacitive Touch	Segment LCD Driver	AES 128-bit
STM32L100 Value line	32 to 256	4 to 16	2	-	-	-	-	-	Up to 8 x 28	-
STM32L151	32 to 512	16 to 80	4 to 16	SDIO FSMC	•	•	•	•	-	-
STM32L152									Up to 8 x 40	
STM32L162	256 to 512	32 to 80	8 to 16	SDIO FSMC	•	•	•	•	Up to 8 x 40	•

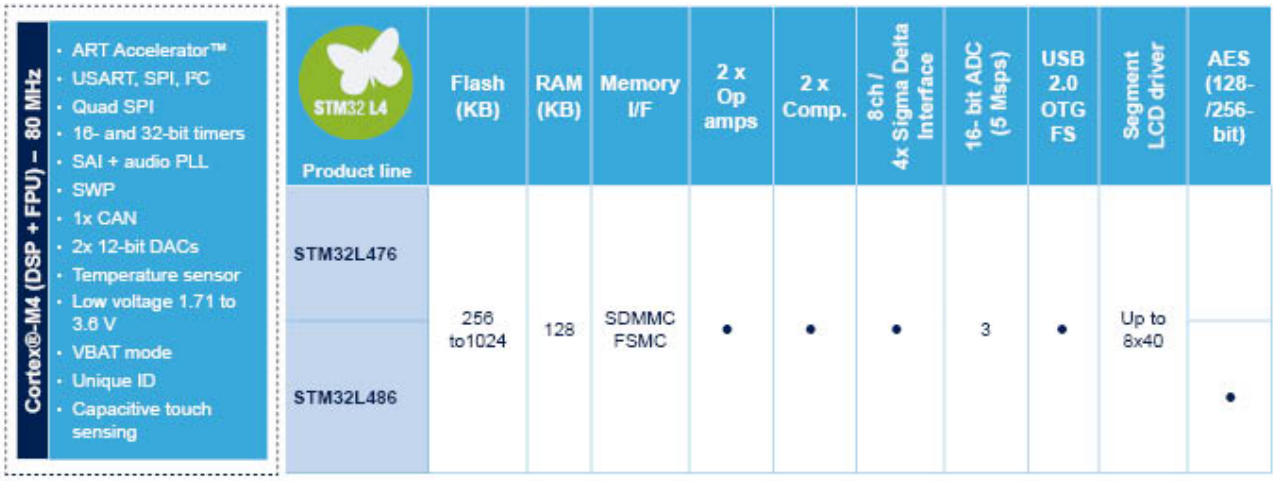
Obr. 2 Řada MCU STM32 L1[STMicroelectronics]

STM32 L4

Série STM32 L4 je nejvyšší řadou mikrokontrolerů s nízkou spotřebou. V testu EEMBC™ ULPBench®, který srovnává mikrokontrolery s nízkou spotřebou, dosáhl skóre 153, což je světový rekord a co se týče výkonu dosahuje 100 DMIPS díky jádru ARM® Cortex®-M4 s FPU a ST ART Accelerator™ na 80 MHz.

STM L4 nabízí dynamické škálování napětí, periférie s nízkou spotřebou jako LP, UART nebo LP časovače dostupné ve STOP módu. Dále pak bezpečnostní a zabezpečovací prvky, početné množství chytrých periférií, moderní analogové periférie s nízkou spotřebou jako operační zesilovače, komparátory, LCD, 12-bit DAC, 16-bit ADC.

Série je dostupná ve dvou řadách STM32L476 (USB, LCD) a STM32L486 (USB, LCD, AES).



Obr. 3 Řada MCU STM32 L4[STMicroelectronics]

2.2 Procesory řady standart

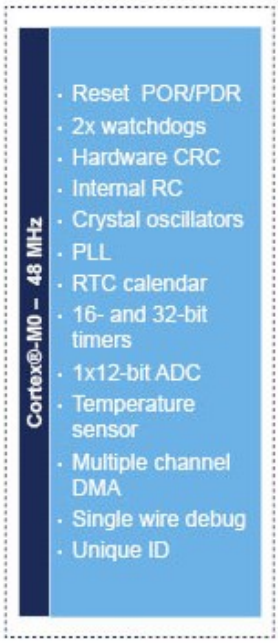
Mezi mikrokontrolery řady standart patří: STM32 F0, STM32 F1 a STM32 F3.

STM32 F0

V mikrokontrolerech STM32 F0 je použito jádro ARM® Cortex®-M0. Tyto MCU přináší 32-bitový výkon a jsou navrženy především pro aplikace, u kterých je kladen důraz na cenu. Mikrokontrolery STM32 F0 kombinují real-time výkon, nízkou spotřebu provozu a pokročilou architekturu a periférie platformy STM32 spolu s jejími základy.

Série STM32 F0 je dostupná ve čtyřech řadách: STM32F0x0 Value line, STM32F0x1 Access line, STM32F0x2 USB line, STM32F0x8 Low voltage line

- STM32F0x0 Value line je úspěšný na trhu 8-bitových a 16-bitových MCU a eliminuje potřebu spravování různých architektur a s nimi spojeného vývoje.
- STM32F0x1 přináší integraci mnoha funkcí a pokrývá široký rozsah velikostí pamětí a balíčků a tím i flexibilitu do zařízení citlivých na cenu.
- STM32F0x2 přináší bohatou konektivitu bezkrystalového USB 2.0 a rozhraní CAN bus, je proto ideální volbou pro komunikační brány, zařízení smart-energy nebo herní terminály.
- STM32F0x8 pracuje s napětím 1,8V ±8% a tak se dobře hodí do přenosných zařízení

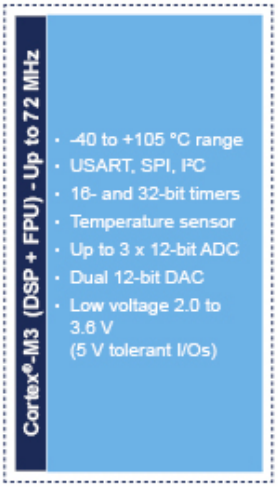


STM32 F0 Product line	FLASH (KB)	RAM (KB)	Power supply	20-byte backup data	12-bit DAC / Comp.	Touch sense	Up to 2xSPI/ I²S, 2xI²C	USART	CEC	CAN	USB
STM32F0x0 Value line	16 to 256	4 to 32	2.4 to 3.6 V				•	6			•
STM32F0x1 Access line	16 to 256	4 to 32	2.0 to 3.6 V	•	• / •	•	•	8	•	•	
STM32F0x2 USB line	16 to 128	4 to 16	2.0 to 3.6 V	•	• / •	•	•	4	•	•	• (crystal-less)
STM32F0x8 Low voltage line	32 to 256	4 to 32	1.8 V +/- 8%	•	• / •	•	•	8	•		• (crystal-less)

Obr. 4 Řada MCU STM32 F0[STMicroelectronics]

STM32 F1

MCU série STM32 F1 jsou mainstreamovou řadou společnosti STM a pokrývají potřeby široké škály průmyslových, lékařských a spotřebitelských aplikací. S touto sérií ST prorazil do světa ARM® Cortex™-M mikrokontrolerů. Vysoký výkon, prvotřídní periférie, nízká spotřeba, operace s nízkým napětím v kombinaci s vysokou úrovní integrace, přijatelnou cenou, jednoduchou architekturou a snadno použitelnými nástroji.



STM32 F1 Product line	FCPU (MHz)	FLASH (bytes)	RAM (KB)	USB 2.0 FS	USB 2.0 FS OTG	FSMC	CAN 2.0B	3-phase MC timer	I²S	SDIO	Ethernet IEEE1588	HDMI CEC
STM32F100 Value line	24	16 K to 512 K	4 to 32			•		•				•
STM32F101	36	16 K to 1 M	4 to 80			•						
STM32F102	48	16 K to 128 K	4 to 16	•								
STM32F103	72	16 K to 1 M	6 to 96	•		•	•	•	•	•		
STM32F105 STM32F107	72	64 K to 256 K	64		•	•	•	•	•		•	

Obr. 5 Řada MCU STM32 F1[STMicroelectronics]

Série se skládá z pěti řad, které jsou pin-to-pin, periférijně a softwarově kompatibilní.

- STM32F100 – 24MHz CPU s řízením motoru a CEC funkcemi
- STM32F101 – 36 MHz CPU, až 1MB Flash
- STM32F102 – 48 MHz CPU s USB FS
- STM32F103 – 72 MHz, až 1MB Flash s řízením motoru, USB a CAN
- STM32F105/107 – 72 MHz CPU s Ethernetem MAC, CAN a USB 2.0 OTG

STM32 F3

STM32 F3 série kombinuje 32-bit ARM® Cortex®-M4 jádro (s FPU a DSP instrukcemi) běžícími na 72MHz s vysokým počtem integrovaných analogových periférií. To vede ke snížení ceny na aplikační úrovni a zjednodušení aplikačního návrhu.

Tato série zahrnuje:

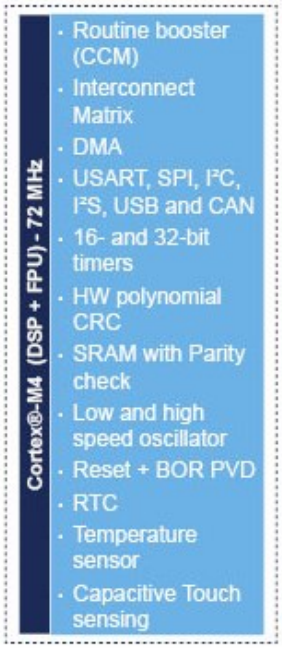
- Ultrarychlé komparátory (25 ns)
- Operační zesilovač s programovatelným přírůstkem
- 12-bit DAC
- Ultrarychlý 12-bit ADC s 5 MSPS (Million Samples Per Second) na každý kanál
- 16-bit sigma-delta ADC s 21 kanály
- Core Coupled Memory SRAM, specifická architektura zvyšující výkon o 43%
- Zdokonalený 144 MHz 16-bit časovač s pulzní modulací
- Časovač s vysokým rozlišením

Vysoká úroveň kompatibility s řadou STM32 F0 garantuje velkou efektivnost při návrhu aplikací s rozdílnou úrovní výkonu.

STM32 F3 série zahrnuje

- STM32F301, STM32F302, STM32F303 jsou produktové řady zahrnující základní, nenákladnou sadu periférií a analogových funkcí schopných spravovat až trojité FOC řízení motoru
- STM32F334 s časovačem s vysokým rozlišením
- STM32F373 s 16-bit sigma-delta ADC pro precizní měřící aplikace
- STM32F3x8 podporujícím 1,8V operace

Provozní teploty jsou od -40 do 85 ° C nebo od -40 do 105 ° C.



STM32 F3 Product line	FLASH (KB)	RAM (KB)	CCM-SRAM	Power supply	ADC 12-bit	ADC 16-bit	12-bit DAC	Fast and Ultra Fast Comp.	Op-Amp (PGA)	Advanced 16-bit PMW Timer	High-Resolution Timer
STM32F301	32 to 64	16		2.0 to 3.6 V	Up to 2		1	3	1	1	
STM32F302	32 to 512	16 to 64		2.0 to 3.6 V	Up to 2		1	Up to 4	Up to 2	1	
STM32F303	32 to 512	16 to 80	•	2.0 to 3.6 V	Up to 4		Up to 3	Up to 7	Up to 4	Up to 3	
STM32F3x4 Digital Power	16 to 64	16	•	2.0 to 3.6 V	2		3	2x Ultra Fast	1	1	• 10ch
STM32F373 Precision measurement	64 to 256	32		2.0 to 3.6 V	1	3	3	2			
STM32F3x8 1.8 V +/-8%	64 to 512	16 to 80	•	1.8 V +/- 8%	Up to 4		Up to 3	Up to 7	Up to 4	Up to 3	

Obr. 6 Řada MCU STM32 F3[STMicroelectronics]

2.3 Procesory řady s vysokým výkonem

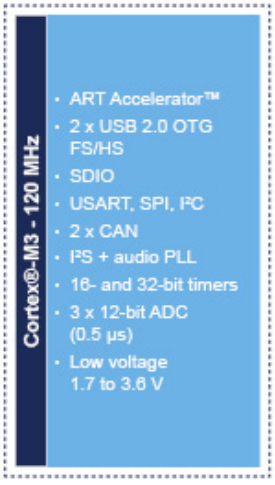
Mezi mikrokontrolery řady s vysokým výkonem patří: STM32 F2, STM32 F4, STM32 F7.

STM32 F2

Řada STM32 F2 se specializuje na vysoký výkon. K tomu jí pomáhá jádro ARM® Cortex™-M3 a použitá 90nm NVM procesní technologie s ART akcelerátorem a vícevrstvou bus maticí. Tato kombinace vytváří jedinečný kompromis mezi cenou a výkonem.

Tato série kombinuje až 1MB Flash paměti, až 128 KB SRAM s Ethernetem, MAC, USB 2.0 HS OTG, rozhraní pro kameru, podpora hardware šifrování a rozraní pro externí paměť.

STM32 F2 jsou schopny dosáhnout až 150 DMIPS/398 CoreMark při 120 MHz $F_{CPU.}$ Série se skládá ze dvou produktových řad, které jsou plně pin-to-pin kompatibilní.



STM32 F2 Product line	FLASH (bytes)	RAM (KB)	Harware Crypto/hash	2 x 12-bit DAC	Ethernet I/F IEEE1588	Camera I/F	FSMC
SM32F215 SM32F205	128 K to 1 M	Up to 128	•	•			•
SM32F217 SM32F207	512 K to 1 M	Up to 128	•	•	•	•	•

Obr. 7 Produktová série STM32 F2[STMicroelectronics]

STM32 F4

Série STM32 F4 používají jádro ARM® Cortex®-M4 a technologii NVM spolu s ART Accelerator™. Ve své třídě dosahuje nejlepších hodnocení a operují s frekvencí 180 MHz při provádění příkazů z Flash paměti.

Spotřeba energie se pohybuje od 100 µA/MHz pro řadu STM32F411 až do 260 µA/MHz pro řadu STM32F439.

STM32 F4 se skládá z osmi kompatibilních produktových řad číslicových regulátorů signálu, které kombinují schopnost řízení v reálném čase (MCU) a výkon při zpracování signálu (DSP).

Řady Advanced:

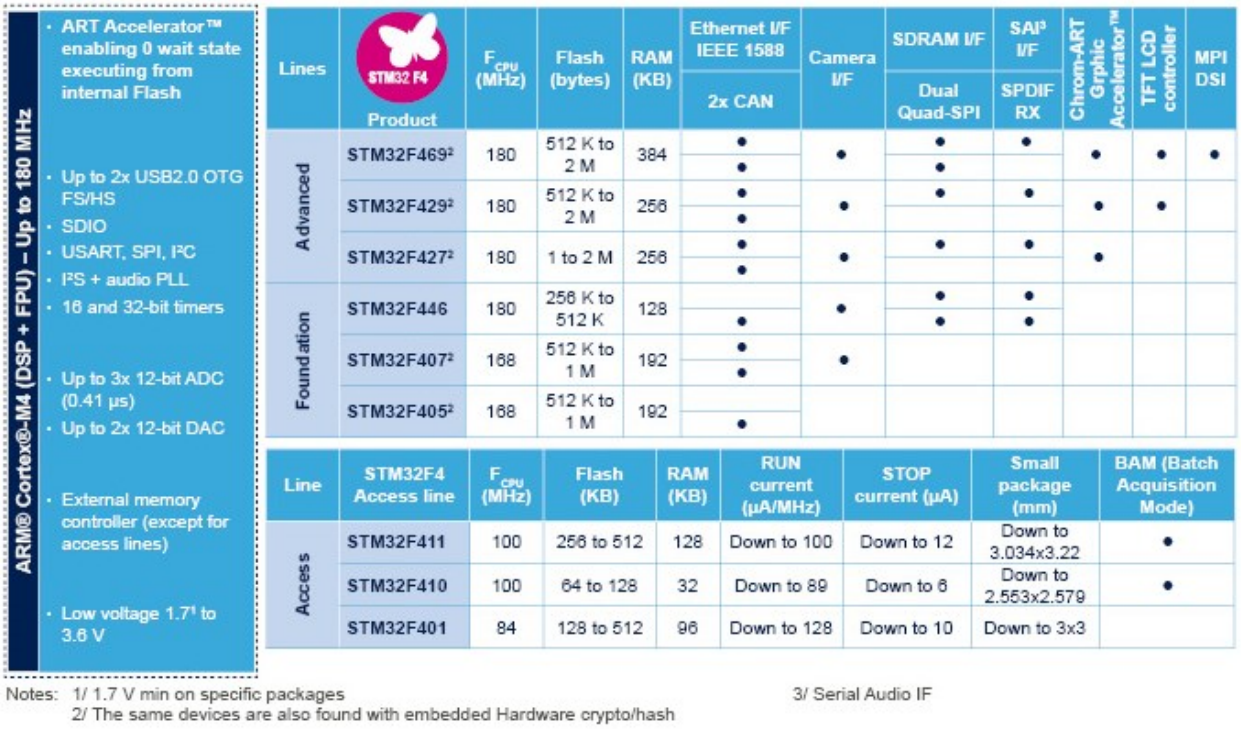
- **STM32F469/479** – 180 MHz CPU/225 DMIPS, až 2 MB dual-bank Flash s SDRAM a QSPI rozhraní, Chrom-ART Accelerator™, LCD-TFT ovladač a MPI-DSI rozhraní
- **STM32F429/439** – 180 MHz CPU/225 DMIPS, až 2 MB dual-bank Flash s SDRAM rozhraním, Chrom-ART Accelerator™ a LCD-TFT ovladač
- **STM32F427/437** – 180 MHz CPU/225 DMIPS, až 2 MB dual-bank Flash s SDRAM rozhraním, Chrom-ART Accelerator™, sériové audio rozhraní, více výkonu a menší statickou spotřebu energie

Řady Foundation:

- **STM32F446** – 180 MHz/225 DMIPS, až 512 KB Flash s dvojitým Quad SPI a SDRAM rozhraní
- **STM32F407/417** – 168 MHz CPU/210 DMIPS, až 1 MB Flash, Ethernet MAC a rozhraní pro kameru
- **STM32F405/415** – 168 MHz CPU/210 DMIPS, až 1 MB Flash s pokročilou konektivitou a šifrováním

Řady Access

- **STM32F411** – 100 MHz CPU/125 DMIPS, vysoká účinnost napájení s novým DMA pro optimalizaci spotřeby pro dávkování dat, velká Flash paměť a RAM, USB, a vylepšený set periférií
- • **STM32F410 -** 100 MHz CPU/125 DMIPS, vysoká účinnost napájení s novým DMA pro optimalizaci spotřeby pro dávkování dat, generátor náhodných čísel, časovač s nízkou spotřebou energie a DAC
- **STM32F401** – 84 MHz CPU/105 DMIPS, nejmenší, cenově přijatelné řešení s nízkou spotřebou energie



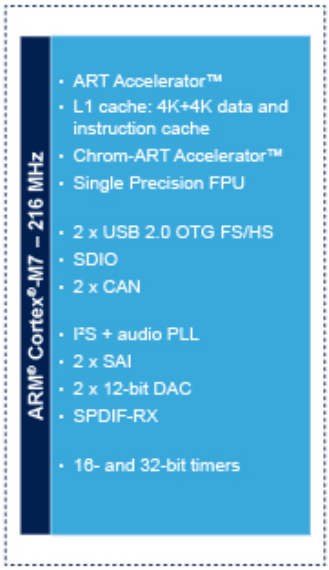
Obr. 8 Produktová série STM32 F4[STMicroelectronics]

STM32 F7

Nejvyšší série produktů STM32 přináší maximální možný teoretický výkon jádra Cortex-M7, bez ohledu na to zda je program vykonáván z vestavěné Flash paměti nebo z externí paměti. 1082 CoreMark /462 DMIPS při 216 MHz fCPU. STM32F7 je pin-to-pin kompatibilní se sérií STM32F4.

Nejnovější jádro Cortex-M7:

- AXI a multi-AHB sběrnice propojuje jádro, periférie a paměti.
- Dva DMA regulátory pro standartní použití a jednoúčelové DMA pro Ethernet, vysokorychlostní USB On-The-Go rozhraní a Chrom-ART grafický akcelerátor
- Rychlost periférií je nezávislá na CPU (možnost změny vnitřních hodin bez zásahu do operací periférií).
- Více periférií, jako dvě sériové audio rozhraní s podporou SPDIF výstupu, tři I²S half-duplex s podporou SPDIF výstupu, dva USB OTG s vyhrazeným napájením a Dual-mode QuadSPI Flash.
- Větší SRAM s rozptýlenou architekturou
- 7 CoreMark / mW při 1.8 V
- Spotřeba 100 µA ve STOP módu, při zachování SRAM a všech ostatních částí
- Zpětná kompatibilita s instrukční sadou Cortex-M4



Product	F_{CPU} (MHz)	Flash (bytes)	RAM (KB)	Ethernet I/F IEEE 1588	Quad SPI	Camera I/F	FMC	TFT LCD controller
STM32F746*	216	512 K to 1 M	320	•	•	•	•	•
STM32F745	216	512 K to 1 M	320	•	•	•	•	

Note: * Crypto/hash hardware on STM32F756 devices only

Obr. 9 Produktová série STM32 F7[STMicroelectronics]

3 Periférie mikrokontrolerů řady STM32F4

Mikrokontrolery používají velké množství periférií a rozhraní, které jim umožňují komunikaci mezi zařízeními nebo řízení vstupních a výstupních signálů. Mezi tyto periférie patří například GPIO (general-purpose input and output), SPI (serial peripheral interface), DMA (DMA controller),DMA2D (chrome-art accelerator™ controller), ADC (analog-to-digital converter), DAC (digital-to-analog converter), DCMI (digital camera interface), LTDC (LCD-TFT controller), TIM (timer), IWDG (independent watchdog), WWDG (window watchdog), RNG (random number generator), UART (universal synchronous asynchronous reciever transmitter), SDIO (secure digital input/output interface), USB nebo Ethernet.

3.1 GPIO(general-purpose input and ouput)

Zkratkou GPIO se označují porty pro vstup a výstup. Díky tomuto rozhraní se zpracovávají digitální a analogové signály, které umožňují mikrokontroleru komunikovat s okolím. Piny jsou rozděleny do 8 portů, které se označují GPIOA až GPIOH, každý z těchto portů obsahuje 16 I/O pinů, které jsou přímo připojeny na sběrnici AHB2 což zajišťuje rychlou odezvu. Každý pin je možné konfigurovat nezávisle na sobě a každý pin může být nastaven jak na vstup (data z externího zdroje jsou přiváděna do MCU) tak na výstup (data jsou odesílána do externích zařízení).

Každý I/O port má čtyři 32-bit konfigurační registry (GPIOx_MODER, GPIOx_OTYPER, GPIOx_OSPEEDR a GPIOx_PUPDR), dva 32-bit datové registry (GPIOx_IDR a GPIOx_ODR), jeden 32-bit set/reset registr (GPIOx_BSRR), jeden 32-bit zamykací registr (GPIOx_LCKR) a dva 32-bit registry pro alternativní volbu funkce (GPIOx_AFRH a GPIOx_AFRL). [STMicroelectronics 2015]

Základní vlastnosti GPIO:

- Řízení až 16 I/O signálů
- Výstupní stav: push-pull nebo open drain + pull-up/down
- Výstupní data z výstupního data registru (GPIOx_ODR) nebo z periférie
- Výběr rychlosti pro každý I/O
- Vstupní stavy: digitální, pull-up/down, analogový
- Vstupní data do vstupního registru (GPIOx_IDR) nebo periférie
- Bit set/reset registr (GPIOx_BSRR) pro bitový přístup do GPIOx_ODR
- Zamykací mechanismus (GPIOx_LCKR) pro uzamčení konfigurace I/O
- Analogová funkce
- Rychlé sepínání schopné změny každé dva hodinové cykly

3.1.1 Popis funkcí GPIO

- Vstup floating

Vstupní pin není připojen k žádnému rezistoru, a tak nemá definovanou žádnou napěťovou úroveň. Toto nastavení pinu není vhodné, protože se na výstupu může zapisovat náhodná hodnota.

- Vstup pull-up

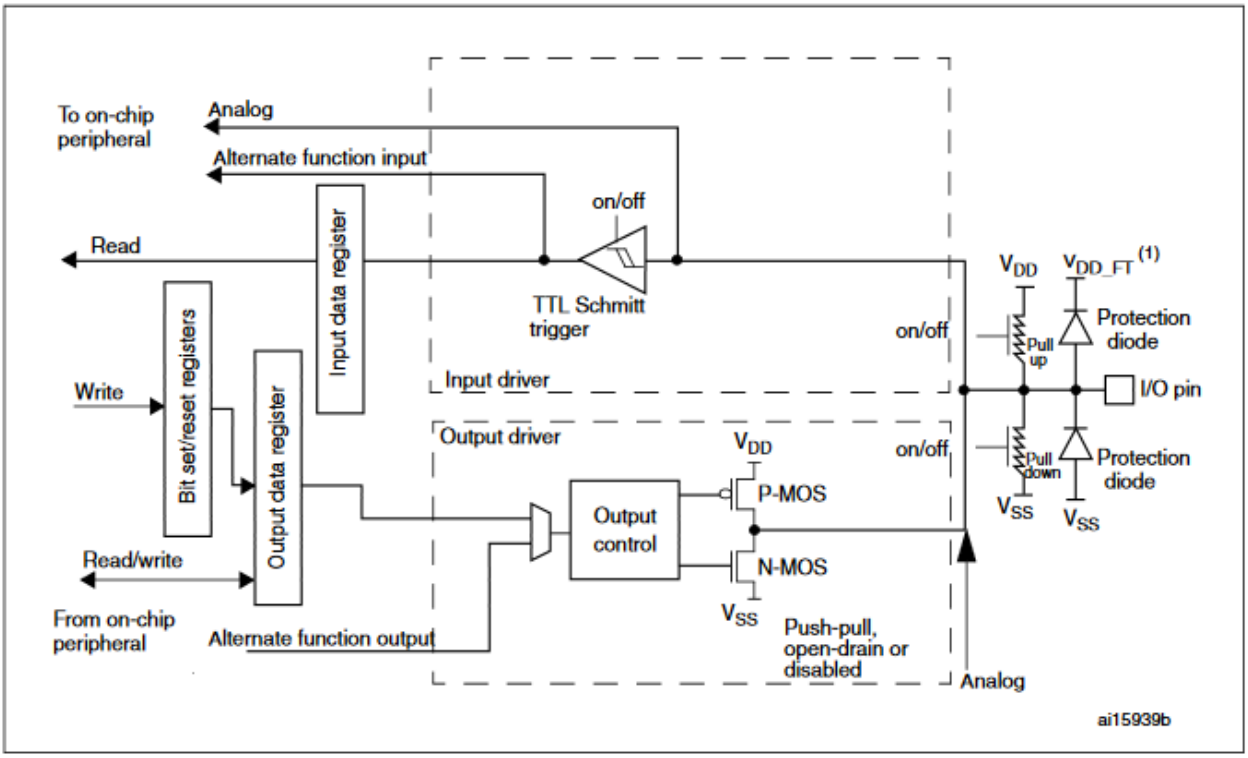
Vstupní pin je připojen ke zdroji, proto je jeho napěťová úroveň vysoko. Při tomto nastavení má pin výchozí hodnotu 1.

- Vstup pull-down

Vstupní pin je uzemněn, proto je jeho napěťová úroveň nízko. Při tomto nastavení má pin výchozí hodnotu 0.

- Výstup open-drain (se schopností pull-up nebo pull-down)
  Open-drain output může pouze uzemnit proud. Výstup může být 0V nebo v druhém případě není výstup ani nízké ani vysoké napětí
- Výstup push-pull (se schopností pull-up nebo pull-down)
  Push-pull výstup může být zdrojem proudu nebo může proud uzemňovat.
- Alternativní funkce open-drain se schopností pull-up nebo pull-down
- Alternativní funkce push-pull se schopností pull-up nebo pull-down
- Analog

Každý I/O port bit je volně programovatelný, avšak k I/O port registru se musí přistupovat jako k 32-bit wordu, half-wordu nebo bytu. Účelem registru GPIOx_BSRR je umožnit atomicky číst/modifikovat přístupy do kteréhokoli GPIO registru. Nevzniká tedy nebezpečí přerušení během čtecím a modifikačním přístupem.



Obr. 10 Schéma I/O port bitu[STMicroelectronics 2015]

I/O port konfigurační registry:

Každý I/O port má čtyři 32-bit konfigurační registry (GPIOx_MODER, GPIOx_OTYPER, GPIOx_OSPEEDR, GPIOx_PUPDR) k nastavení až 16 vstupů/výstupů. GPIOx_MODER registr se používá k výběru módu vstupu/výstupu (input, output, altenative function, analog). GPIOx_OTYPER a GPIOx_OSPEEDR registry se používají k výběru typu (push-pull nebo open-drain) a rychlosti. GPIOx_PUPDR registr se používá k výberu pull-up/pull-down bez ohledu na směr vstupu/výstupu.

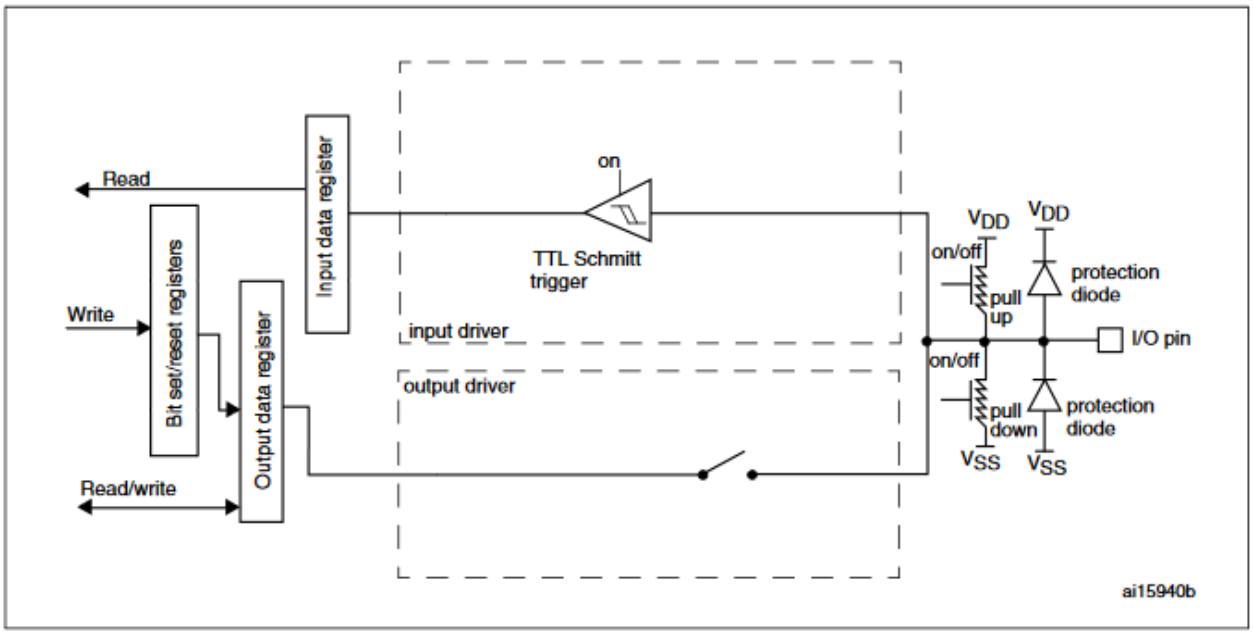
I/O port datové registry:

Každý GPIO port má dva 16-bit datové registry (GPIOx_IDR a GPIOx_ODR). GPIOx_ODR je registr, kde se skladují data k výstupu. GPIOx_ODR je registr určený pro čtení i zápis. Data určená ke vstupu jsou skladována v registru GPIOx_IDR. Tento registr je určen pouze pro čtení.

3.1.2 Konfigurace GPIO při vstupním nastavení

Jakmile je I/O port nastaven jako vstup:

- Výstupní buffer je zakázán
- Vstup Schmittova klopného obvodu je aktivován
- Pull-up a pull-down rezistory jsou aktivovány na základě hodnoty v registru GPIOx_PUPDR
- Data, která jsou na I/O pinu jsou vzorkována do vstupního registru každý AHB1 cyklus
- K přístupu dat ke čtení slouží I/O State

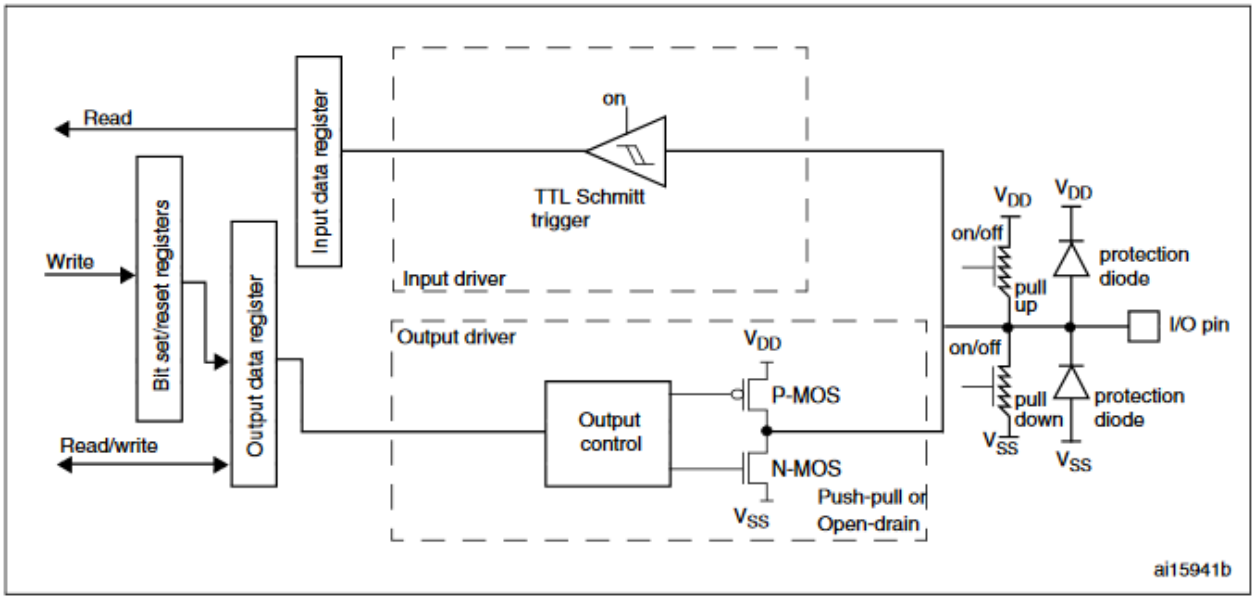


Obr. 11 Nastavení floating/pull up/pull down vstupu[STMicroelectronics 2015]

3.1.3 Konfigurace GPIO při výstupním nastavení

Jakmile je I/O port nastaven jako výstup:

- Výstupní buffer je povolen
- Vstup Schmittova klopného obvodu je aktivován
- Slabé pull-up a pull-down rezistory jsou nebo nejsou aktivovány na základě hodnoty v registru GPIOx_PUPDR
- Data, která jsou na I/O pinu jsou vzorkována do výstupního registru každý AHB1 cyklus
- Přístup ke čtení registru vstupních dat dostane I/O stav
- Přístup ke čtení registru výstupních dat dostane poslední zapsaná hodnota



Obr. 12 Nastavení výstupu[STMicroelectronics 2015]

3.2 SPI (serial peripheral interface)

SPI rozhraní zajišťuje dvě hlavní funkce, a to podporu protokolu SPI nebo I^2S audio protokolu. Jako výchozí protokol je nastaven SPI, nicméně je možné přepnout rozhraní ze SPI na I^2S pomocí softwaru.

Sériové periferní rozhraní (SPI) umožňuje half/full-duplex, synchronní sériovou komunikaci mezi externími zařízeními. Rozhraní může být nastaveno jako master. V tomto případě poskytuje komunikační hodiny (SCK) externímu slave zařízení. Rozhraní může také pracovat v multimaster módu.

Rozhraní může být použito k mnoha účelům včetně simplexního synchronního přenosu na dvou linkách s možností obousměrné datové linky nebo spolehlivou komunikaci užitím cyklického redundantního součtu (CRC).

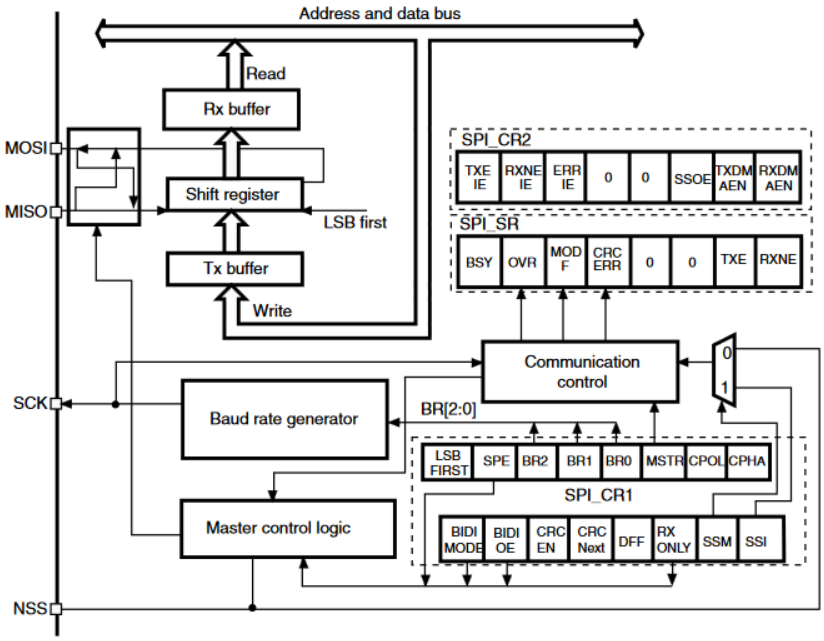
I^2S je také synchronní sériové komunikační rozhraní. Může adresovat čtyři různé audio standardy: I^2S Phillps standard, MSB-justified standard, LSB-justified standard, PCM standard. Může operovat jako master nebo slave zařízení ve full-duplex módu (užitím 4 pinů) nebo half-duplex módu (užitím 3 pinů). Při nastavení I^2S jako master může toto rozhraní poskytovat komunikační hodiny externímu slave zařízení.

Vlastnosti SPI:

- Full-duplex synchronní přenos na třech linkách
- Výběr 8-bit nebo 16-bit datového rámce
- Master nebo slave operace
- Multimaster mód
- Osm baud rate prescalerů v master módu
- Slave mód frekvence
- NSS hardwarový nebo softwarový management
- Programovatelná polarita a fáze hodin
- Flagy pro vysílání a přijímání s možností přerušení

Vlastnosti I^2S:

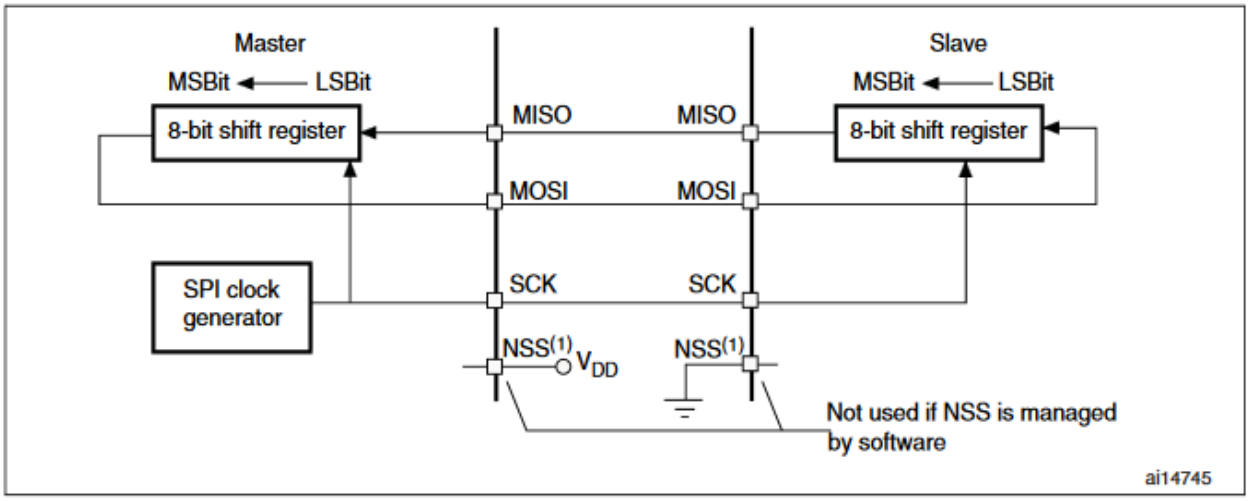
- Full duplex komunikace
- Half duplex komunikace (pouze jako vysílač nebo přijímač)
- Master nebo slave operace
- Formát dat může být 16-bit, 24-bit nebo 32-bit
- Packetový rámec je fixní 16-bit nebo 32-bit podle audio kanálu
- Programovatelná polarita hodin
- Podporované protokoly: I^2S Phillps standard, MSB-justified standard, LSB-justified standard, PCM standard



Obr. 13 Blokový diagram SPI [STMicroelectronics 2015]

Obvykle je SPI připojeno k externímu zařízení čtyřmi piny:

- MISO: Master In / Slave Out data. Tento pin může být použit pro vysílač v slave módu a přijímač v master módu.
- MOSI: Master Out / Slave In data. Tento pin může být použit pro přijímač v slave módu a vysílač v master módu.
- SCK: Serial Clock výstup pro SPI masters and vstup pro SPI slaves
- NSS: Slave Select. Toto je volitelný pin pro výběr slave zařízení. Tento pin se chová jako ‘chip select’, aby nechal SPI master zařízení komunikovat se slave zařízeními individuálně a vyhnul se kolizi na datových linkách. NSS vstupy mohou být řízeny standartními IO porty na master zařízení. NSS pin může být také použit jako výstup pokud je povolen (SSOE bit) a řízen nízko pokud je SPI v master módu. Při tomto chování všechny NSS piny ze zařízení připojených k master NSS pinu vidí nízký stav a stanou se slave zařízení jakmile jsou nakonfigurovány v NSS hardware módu. Pokud jsou nakonfigurovány jako master módu s NSS nastaveným jako vstup a pokud je NSS zataženo nízko, SPI vstoupí do chybového master mód stavu: MSTR bit je vymazán a zařízení přejde do módu slave.



Obr. 14 Single master/ single slave aplikace[STMicroelectronics 2015]

MOSI piny jsou připojeny k sobě a MISO piny jsou také připojeny k sobě. V tomto směru jsou data přenášena sériově mezi master a slave zařízením (nejvíce významový bit první).

Komunikace je vždy zahájena master zařízením. Jakmile master zařízení vyšle data ke slave zařízení přes MOSI pin, slave zařízení odpoví přes MISO pin. Z toho vyplývá full-duplex komunikace se synchronizovanými vstupními a výstupními daty pomocí stejného hodinového signálu (který je poskytován master zařízením přes SCK pin).

3.2.1 Nastavení SPI do slave módu:

Ve slave nastavení, signál sériových hodin je přijímaný na SCK pin z master zařízení. Hodnota nastavená na BR[2:0] bitech v registru SPI_CR1 neovlivní přenosovou rychlost dat.

Postup nastavení SPI do slave módu:

1. Nastavte DFF bit pro definování 8-bit nebo 16-bitového datového rámce
2. Zvolte CPOL a CPHA bity pro definování jednoho ze čtyř vztahů mezi přenosem dat a sériovými hodinami. Pro správný přenos dat CPOL a CPHA musí být nastaveny stejným směrem v master a slave zařízení. Tento krok není vyžadován, pokud je vybraný TI mód přes FRF bit v registru SPI_CR2.
3. Formát rámce (nejvíce významový bit první nebo nejméně významový bit první podle hodnoty v SPI_CR1 registru) musí být stejný jako master zařízení. Tento krok není vyžadován, pokud je vybrán TI mód.
4. V hardware módu musí být NSS pin připojen na nízko-úrovňový signál během dokončení bytové sekvence vysílání. V NSS software módu nastavte SSM bit a vyprázdněte SSI bit v registru SPI_CR1. Tento krok není vyžadován, pokud je nastaven TI mód.
5. Nastavte FRF bit v registru SPI_CR2 pro výběr TI mode protokolu pro sériovou komunikaci.
6. Vyprázdněte MSTR bit a nastavte SPE bit (oba v SPI_CR1 registru) pro přiřazení pinu k alternativním funkcím.

V tomto nastavení je MOSI pin datový vstup a MISO datový výstup.

Vysílací sekvence:

Datový byte je paralelně nahrán do Tx bufferu během zapisovacího cyklu.

Vysílací sekvence začíná jakmile slave zařízení příjme hodinový signál a nejvíce významový bit dat na jeho MOSI pin. Zbývající bity (7 bitů při 8-bit datovém rámci, 15 bitů při 16-bit datovém rámci) jsou nahrány do shift registru. TXE flag v SPI_SR registru je nastaven na data transfer z Tx bufferu do shift registru a přerušení je generováno pokud je nastaven TXEIE bit v registru SPI_CR2.

Přijímací sekvence:

Pro přijímač, pokud je datový transfer dokončen:

- Data v shift registru jsou přesunuta do Rx bufferu a RXNE flag (registr SPI_SR) je nastaven.
- Přerušení je generováno, jakmile je nastaven RXNEIE bit v registru SPI_CR2

Při poslední hraně vzorkovacích hodin je nastaven RXNE bit, kopie datového bytu přijatého do shift registru je přesunuta do Rx bufferu. Jakmile je přečten SPI_DR registr, SPI vrátí hodnotu v bufferu.

Vymazání RXNE bitu je provedeno přečtením registru SPI_DR.

SPI TI protokol ve slave módu:

Ve slave módu je SPI rozhraní kompatibilní s TI protokolem. FRF bit registru SPI_CR2 může být použit k nastavení slave SPI sériových komunikací, aby byly kompatibilní s tímto protokolem.

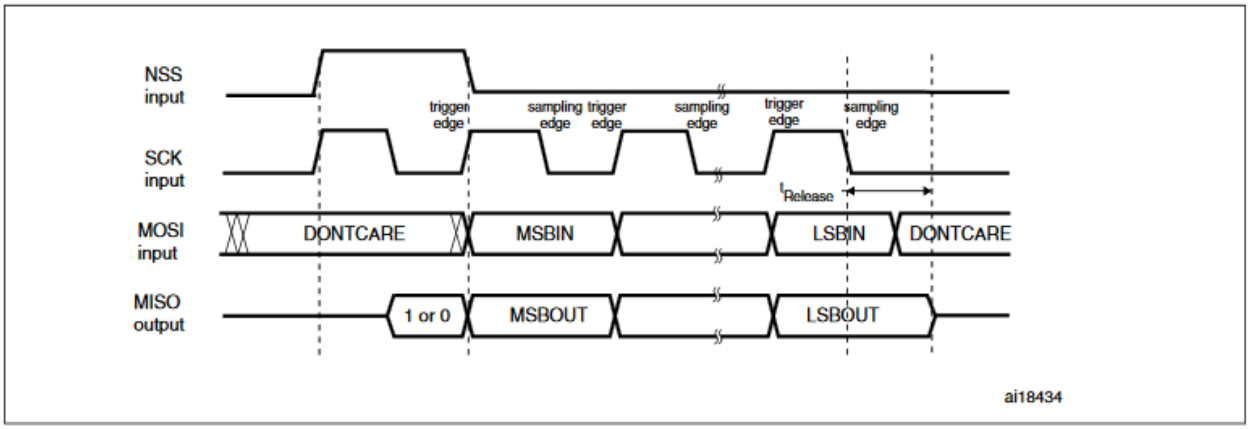
Polarita hodin a fáze jsou donuceny k přizpůsobení se požadavkům protokolu TI bez ohledu na hodnoty zapsané v registru SPI_CR1. Správa NSS je také specifická při použití protokolu TI, což činí nastavení správy NSS přes registry SPI_CR1 a SPI_CR2 transparentní pro uživatele.

Ve slave módu je násobič přenosové rychlosti použit k řízení okamžiku kdy MISO pin změní stav na HI-Z. Jakákoli přenosová rychlost může být použita, což umožňuje určit tento moment s optimální flexibilitou. Avšak přenosová rychlost je obvykle nastavena dle externí přenosové rychlosti hodin. Čas, kdy se MISO signál stal HI-Z($t_{release}$) závisí na vnitřních resynchronizačních hodnotách přenosové rychlosti nastavené na bitech BR[2:0] registru SPI_CR1 dle vztahu:

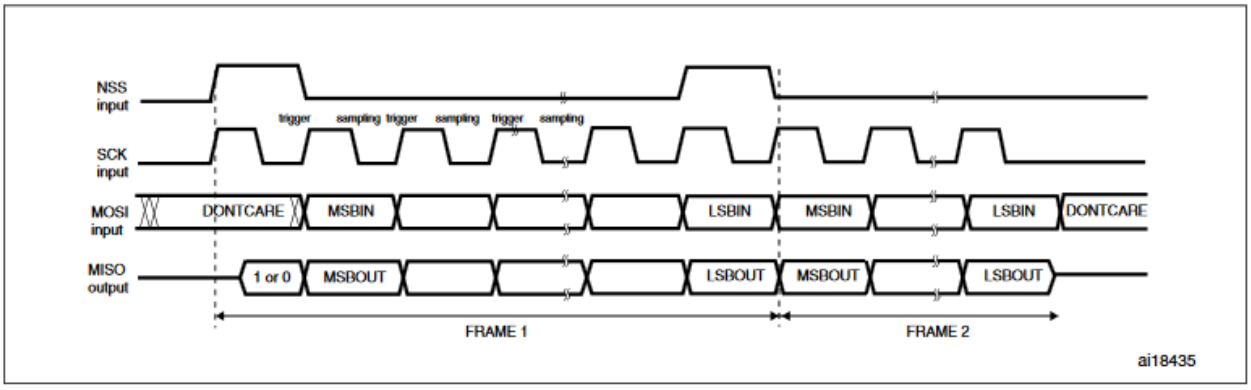
$$\frac{t_{baud\_rate}}{2} + 4 \times t_{pclk} < t_{release} < \frac{t_{baud\_rate}}{2} + 6 \times t_{pclk}$$

, kde t_{baud_rate} - čas hodin násobiče přenosové rychlosti

t_{pclk} - čas periférních hodin



Obr. 15 TI mode - Slave mode, single transfer[STMicroelectronics 2015]



Obr. 16 TI mode - Slave mode, continuous transfer[STMicroelectronics 2015]

3.2.2 Nastavení SPI do master módu:

V master konfiguraci jsou sériové hodiny generovány na pin SCK.

Postup nastavení SPI do slave módu:

1. Vyberte bity BR[2:0] pro definici přenosové rychlosti.
2. Vyberte CPOL a CPHA bity pro definici jednoho ze čtyř vztahů mezi přenosem dat a sériovými hodinami. Tento krok není vyžadován, pokud je vybrán TI mód.
3. Nastavte DFF bit pro definici 8-bit nebo 16-bit datového rámce.
4. Nastavte LSBFIRST bit v registru SPI_CR1 pro definici rámcového formátu. Tento krok není vyžadován, pokud je vybrán TI mód.
5. Pokud je vyžadován NSS bit při vstupním nastavení, v hardware módu připojte NSS pin na vysoko-úrovňový signál během dokončení bytové sekvence vysílání. V NSS software módu nastavte SSM a SSI bity v registru SPI_CR1. Pokud je NSS pin vyžadován při výstupním nastavení, měl by být nastaven pouze SSOE bit. Tento krok není vyžadován, pokud je vybrán TI mód.
6. Nastavte FRF bit v registru SPI_CR2 pro výběr TI protokolu pro sériovou komunikaci
7. MSTR a SPE bity musí být nastaveny.

V tomto nastavení je MOSI pin datový výstup a MISO pin datový vstup.

Vysílací sekvence:

Vysílací sekvence začne, jakmile je byte zapsán do Tx bufferu.

Datový byte je paralelně nahrán do shift registru (z interní sběrnice) během přenosu prvního bitu a pak je sériově posunut na MOSI pin (nejvíce významový bit první nebo nejméně významový bit první v závislosti na LSBFIRST bitu v registru SPI_CR1). TXE flag je nastavena na transfer of data z Tx bufferu do shift registru a přerušení je generováno pokud je nastaven TXEIE bit v registruSPI_CR2.

Přijímací sekvence:

Pro přijímač, kdy je dokončen datový transfer:

- Data v shift registru jsou přenesena do RX bufferu a RXNE flag je nastaven.
- Přerušení je generováno, pokud je nastaven RXNEIE bit v registru SPI_CR2.

Při poslední hraně vzorkovacích hodin je nastaven RXNE bit, kopie datového bytu přijatého v shift registru je přesunuta do Rx bufferu. Jakmile je přečten registr SPI_DR, SPI vrátí hodnotu v bufferu.

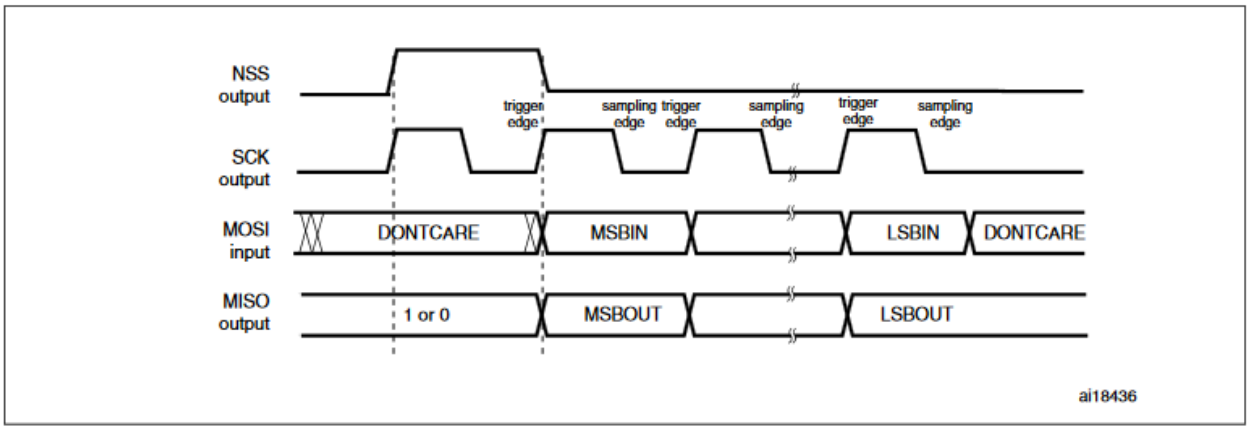
Vymazání RXNE bitu je provedeno přečtením registru SPI_DR.

Nepřetržitý přenosový proud může být udržován, pokud jdou data k dalšímu přenosu vložena do Tx bufferu, jakmile přenos začne. TXE flag by měl být nastaven na 1 před jakýmkoli pokusem zapsat do Tx bufferu.

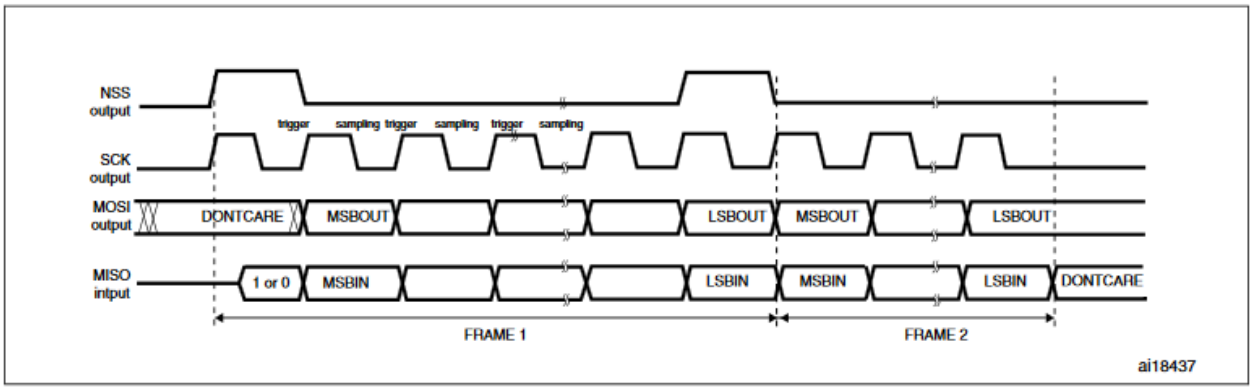
SPI TI protokol v master módu:

V master módu je SPI rozhraní kompatibilní s TI protokolem. FRF bit registru SPI_CR2 může být použit k nastavení SPI sériových komunikací pro kompatibilitu s tímto protokolem.

Polarita hodin a fáze jsou donuceny k přizpůsobení se požadavkům protokolu TI bez ohledu na hodnoty zapsané v registru SPI_CR1. Správa NSS je také specifická při použití protokolu TI, což činí nastavení správy NSS přes registry SPI_CR1 a SPI_CR2 transparentní pro uživatele.



Obr. 17 TI mode - master mode, single transfer[STMicroelectronics 2015]



Obr. 18 TI mode - master mode, continuous transfer[STMicroelectronics 2015]

3.2.3 Nastavení SPI pro half-duplex komunikaci:

SPI je schopno pracovat v half-duplex módu ve dvou konfiguracích.

- 1 clock a 1 obousměrná datová linka (BIDIMODE=1)

  Tento mód je povolen nastavením BIDIMODE bitu v registru SPI_CR1. V tomto módu SCK je použit pro hodiny a MOSI v master nebo MISO ve slave módu je použit pro datovou komunikaci. Přenosový směr (vstup/výstup) je vybrán BIDIOE bitem v registru SPI_CR1. Pokud je nastaven na 1 je datová linka výstupem v opačném případě vstupem.
- 1 clock a 1 datová linka (transmit-only nebo recieve-only) (BIDIMODE=1)

  V tomto módu může aplikace používat SPI buď v transmit-only módu nebo recieve-only módu.
  - Transmit-only mód je podobný jako full-duplex mód: data jsou přenesena na vysílací pin (MOSI pin v master módu nebo MISO pin ve slave módu) a přijímačový pin (MISO pin v master módu nebo MOSI pin ve slave módu) může být použit jako pin standartní vstup/výstup. V tomto případě potřebuje aplikace ignorovat Rx buffer (pokud je datový registr přečten neobsahuje přijatou hodnotu).

- V reciever-only módu může aplikace zakázat SPI výstupní funkci nastavením RXONLY bitu v registru SPI_CR2. V tomto případě je uvolněn vysílací IO pin (MOSI v master módu nebo MISO ve slave módu), takže může být použit v jiným účelům.

Pro začátek komunikace v recieve-only módu nastavte a povolte SPI:

- V master módu začne komunikace ihned a skončí jakmile je vyčištěn SPE bit a současný příjem skončí. V tomto módu není nutnost číst BSY flag. Je vždy nastaven jakmile probíhá SPI komunikace.
- Ve slave módu SPI pokračuje v přijímáni tak dlouho dokud je NSS stažen dolů (nebo dokud není vyčištěn SSI bit v NSS software módu) a SCK je aktivní

3.3 USART(universal synchronous asynchronous reciever transmitter)

Univerzální synchronní asynchronní vysílač a přijímač nabízí flexibilní způsob full – duplex výměnu dat s externími zařízeními, který vyžaduje průmyslový standart NRZ asynchronní sériový data formát. USART nabízí široký rozsah přenosových rychlostí užitím baud rate generátor.

Podporuje synchronní jednosměrnou komunikaci a half-duplex komunikaci po jednom vodiči. Také podporuje LIN, SmartCard Protocol a IrDA SIR ENDEC specifikace a moderní operace jako CTS/RTS. Umožňuje také multiprocesorovou komunikaci. [STMicroelectronics(2)]

Vlastnosti USART:

- Full duplex, asynchronní komunikace
- NRZ standart formát
- Konfigurovatelný oversampling
- Programovatelná délka slova (8 nebo 9 itů)
- Konfigurovatelné stop bity – podpora pro 1 nebo 2 stop bity
- IrDA SIR encoder decoder
- Half-duplex komunikace po jednom vodiči

Popis funkcí USART:

Rozhraní je externě připojeno k dalšímu zařízení třemi piny. Jakákoli USART obousměrná komunikace vyžaduje minimálně 2 piny: Recieve Data In (RX) a Transmit Data Out (TX):

RX: Recieve Data Input je sériový datový vstup. Převzorkovací techniky jsou použity k odlišení platných dat od šumu.

TX: Transmit Data Output. Jakmile je vysílač zakázán výstupní pin se vrátí do jeho I/O port konfigurace. Jakmile je vysílač povolen a nemá nic k vysílání je TX pin na vysoké úrovni. V single-wire a smartcard módech je tento pin použit k vysílání a přijímání dat (na USART úrovni jsou data přijímána na SW_RX).

Přes tyto piny jsou sériová data vysílána a přijímána v normal USART módu jako rámce, které obsahují:

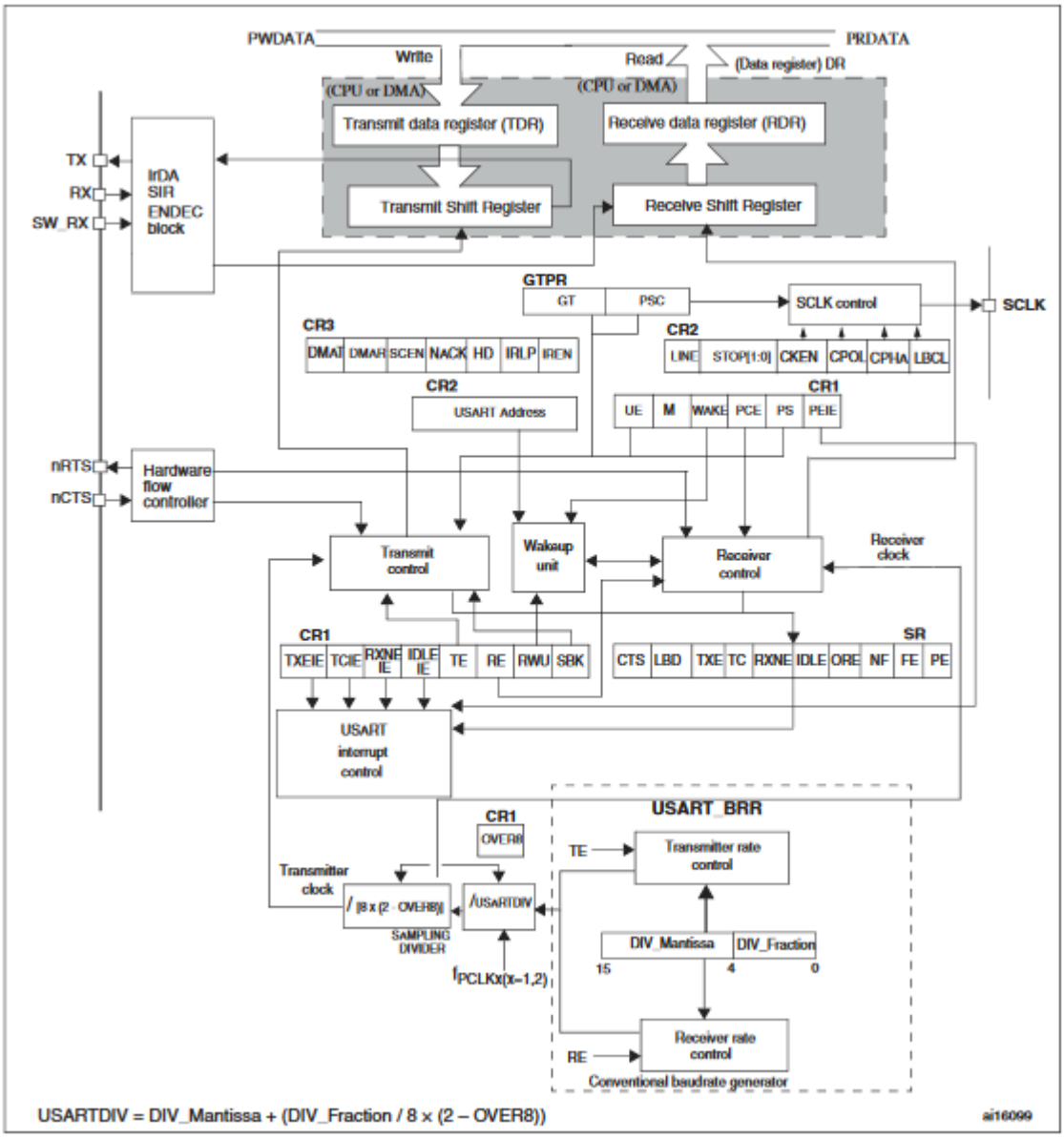
- Předchozí činnost linky pro vysílání nebo přijímání
- Start bit
- Datové slovo (8 nebo 9 bitů) s prvním nejméně významovým bitem
- 0,5;1;1,5;2 stop bity indikující, že rámec je kompletní
- Toto rozhraní používá fractional generátor přenosové rychlosti s 12-bit mantisou a 4-bit exponentem
- Status registr (USART_SR)
- Datový registr (USART_DR)
- Baud rate registr (USART_BRR) - 12-bit mantisa a 4-bit exponent
- Guardtime registr (USART_GTPR) pro případ Smartcard módu

Následující pin je vyžadován pro nastavení v synchronním módu:

- **SCLK:** Výstup vysílacích hodin. Tento pin je výstupem vysílacích datových hodin pro synchronní vysílání odpovídající SPI master módu (žádné pulzy hodin na start bitu a stop bitu, a softwarová možnost poslat pulz hodin na posledním datovém bitu). Paralelně mohou být data přijímána synchronně na RX. To může být použito k ovládání periférií, které mají shift registry (např. LCD drivery). Fáze a polarita hodin jsou softwarově programovatelné. Ve smartcard módu může SCLK poskytovat hodiny pro smartcard.

Následující piny jsou vyžadovány v Hardware flow control módu:

- **nCTS:** Clear To Send blokuje datový přenos na konci aktuálního přenosu (pokuj je na vysoké úrovni)
- **nRTS:** Request To Send indikuje, že USART je připraven k příjmu dat (pokud je na nízké úrovni)



Obr. 19 Blokový diagram periférie USART[STMicroelectronics 2015]

3.3.1 Vysílač

Vysílač může posílat datová slova buď jako 8-bit nebo 9-bit v závislosti na stavu M bitu. Jakmile je nastaven bit povolení přenosu (TE) jsou data ze shift registru výstupem na TX pinu a odpovídající pulzy hodin jsou výstupem na SCLK pinu.

Přenos znaku:

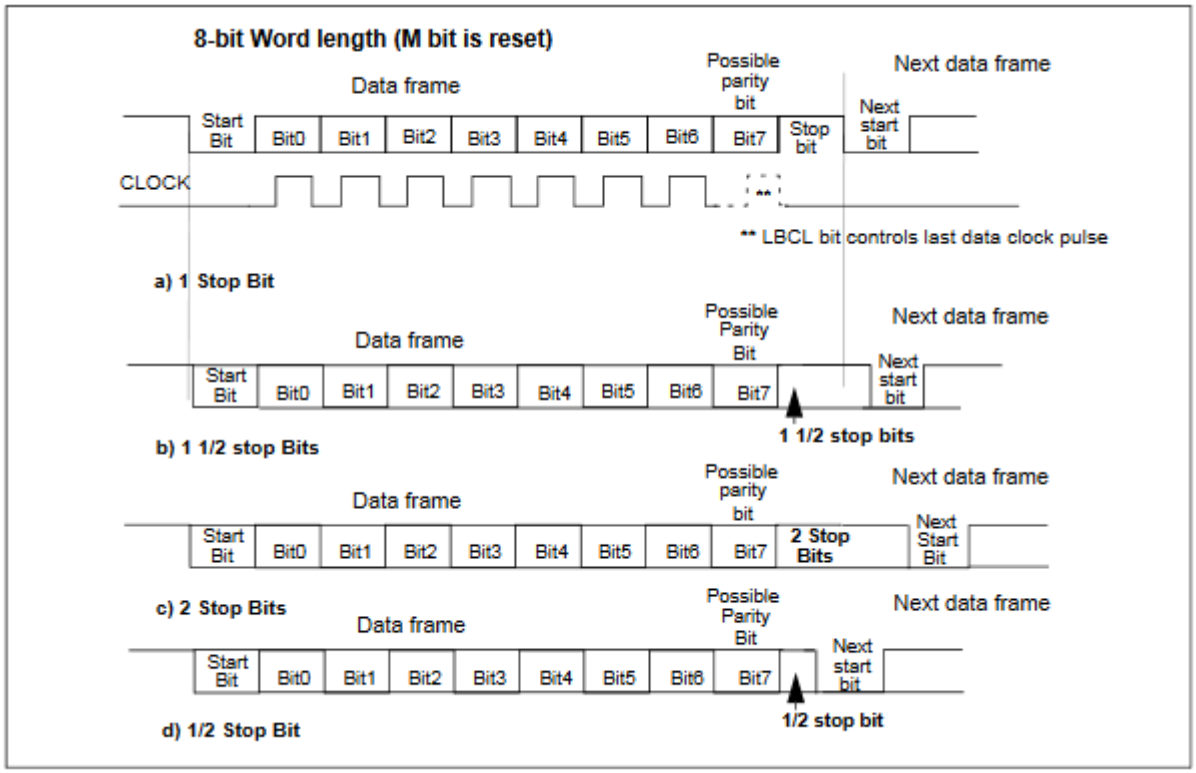
Během USART vysílání jsou data posouvána na TX pin (nejméně významový bit jako první). V tomto módu se registr USART_DR skládá z bufferu (TDR) mezi interní sběrnicí a přenosovým shift registrem.

Každý znak je předcházen start bitem, který je na nízké logické úrovni pro jednu bit periodu. Znak je ukončen konfigurovatelným počtem stop bitů.

Následující stop bity jsou podporovány rozhraním USART: 0,5; 1; 1,5 a 2 stop bity.

Konfigurovatelné stop bity:
Počet stop bitů, které se budou vysílat s každým znakem, může být naprogramován v Control registru 2 bity 13, 12.

- **1 stop bit:** Výchozí hodnota počtu stop bitů.
- **2 stop bity:** Toto je podporováno normal USART, single-wire a modem módem.
- **0,5 stop bitu:** Používá se pro příjem dat ve smartcard módu
- **1,5 stop bitu:** Používá se při vysílání a příjem dat ve smartcard módu



Obr. 20 Konfigurovatelné stop bity[STMicroelectronics 2015]

Postup:

1. Povolte USART zapsáním 1 na UE bit v registru USART_CR1.
2. Nastavte M bit v registru USART_CR1 pro definování délky slova.
3. Nastavte počet stop bitů v registru USART_CR2.
4. Povolte DMA v registru USART_CR3 pokud budete používat Multi buffer Komunikaci. Nakonfigurujte DMA registr.
5. Vyberte požadovanou přenosovou rychlost použitím USART_BRR registru.
6. Nastavte TE bit v USART_CR1 pro odeslání nečinného rámce jako první přenos.
7. Zapište data k odeslání do registru USART_DR (toto vyčistí TXE bit). Toto opakujte pro každá data k přenesení v případě použití single bufferu.
8. Po zapsání posledních dat do USART_DR registru počkejte dokud TC=1. To indikuje, že přenos posledního rámce byl dokončen. Toto je vyžadováno pro případ, kdy je USART zakázán nebo vstoupí do Halt módu pro zajištění bezchybnosti posledního přenosu.

Single byte komunikace:

Vyčištění TXE bitu je vždy provedeno zápisem do datového registru.

TXE bit je nastaven hardwarem a indikuje:

- Data byla přesunuta z TDR do shift registru a datový přenos začal.
- TDR registr je prázdný.
- Další data mohou být zapsány do USART_DR registru bez přepsání předchozích dat.

Tento flag generuje přerušení, pokud je nastaven TXEIE bit.

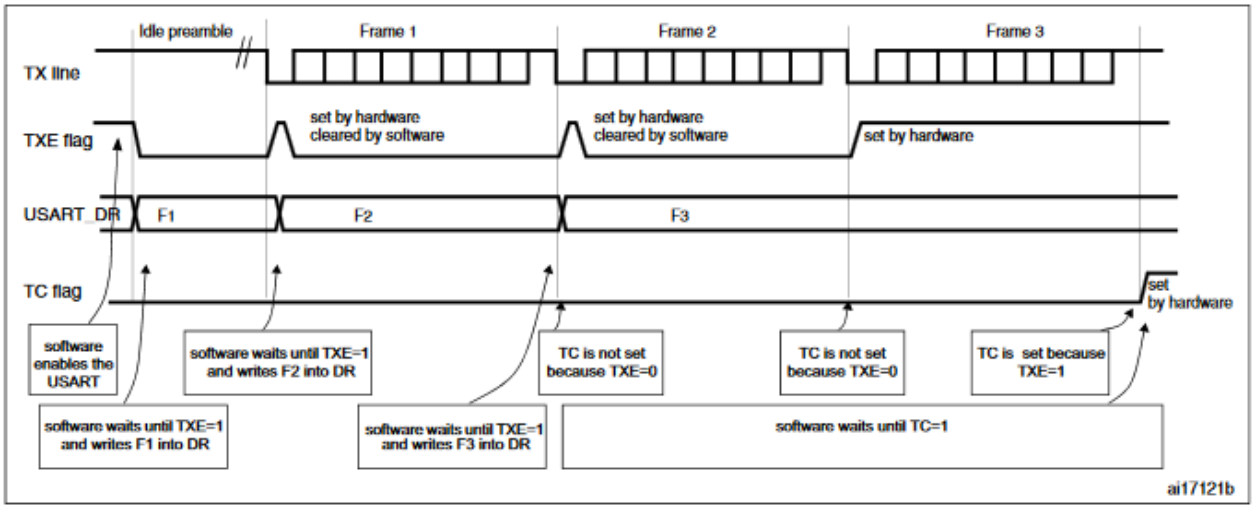
Jakmile probíhá přenos, instrukce zapsání do registru USART_DR uskladní data v TDR registru. Tyto data jsou zkopírována do shift registru na konci aktuálního přenosu.

Pokud přenos neprobíhá, instrukce zapsání do USART_DR registru vloží data přímo do shift registru, datový přenos začne a TXE bit je okamžitě nastaven.

Pokud je rámec přenesen a TXE bit je nastaven, TC bit se nastaví do vysoké úrovně. Přerušení je generováno, pokud je TCIE bit nastaven v registru USART_CR1.

TC bit je vyčištěn těmito softwarovými sekvencemi:

1. Čtením z registru USART_SR
2. Zápisem do registru USART_DR



Obr. 21 Chování TC/TXE během přenosu[STMicroelectronics 2015]

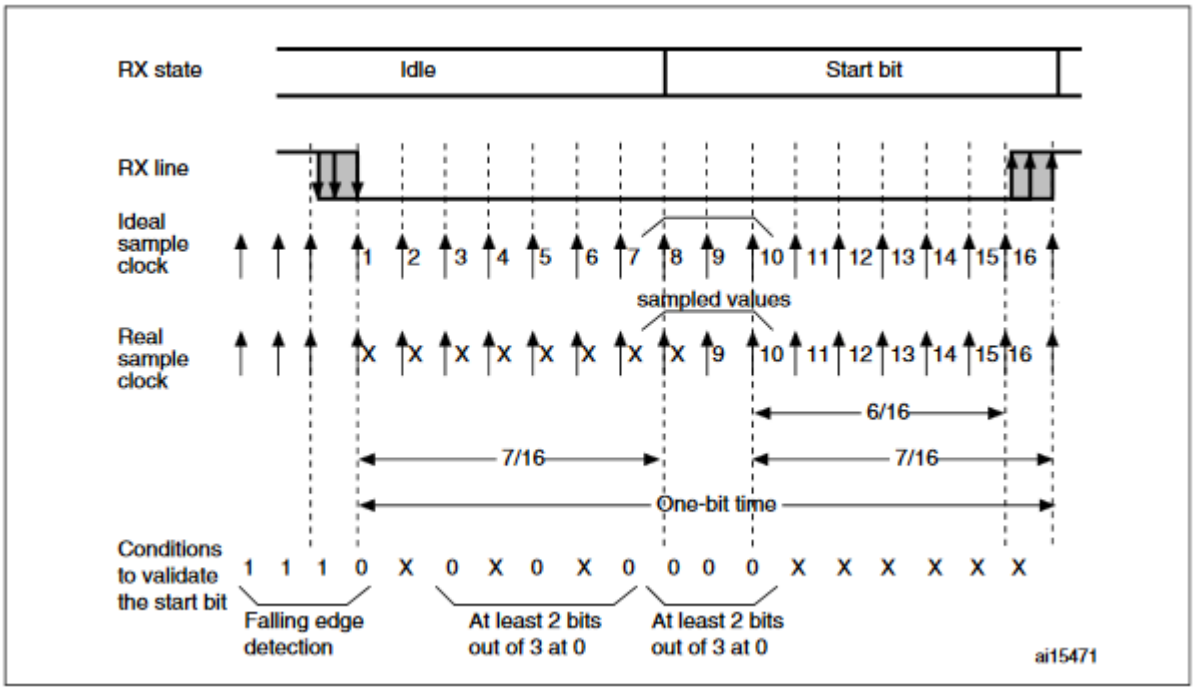
3.3.2 Přijímač

USART může přijímat buď 8-bit nebo 9-bit slova v závislosti na M bitu v registru USART_CR1.

Detekce start bitu:

Sekvence start bit detekce je stejná když vzorkujeme po 16 nebo po 8.

V USART rozhraní, start bit je detekován, jakmile specifická sekvence vzorku je rozpoznána. Tato sekvence je: 1 1 1 0 X 0 X 0 X 0 0 0 0.



Obr. 22 Detekce start bitu při vzorkování po 16 nebo po 8[STMicroelectronics 2015]

Příjem znaků:

Během USART příjmu jsou data posouvána (nejméně významový bit jako první) přes RX pin. V tomto módu USART_DR registr se skládá z bufferu (RDR) mezi interní sběrnicí a přijatým shift registrem.

Postup:

1. Povolte USART zapsáním 1 na UE bit v registru USART_CR1.
2. Nastavte M bit v registru USART_CR1 pro definování délky slova.
3. Nastavte počet stop bitů v registru USART_CR2.
4. Povolte DMA v registru USART_CR3 pokud budete používat Multi buffer Komunikaci. Nakonfigurujte DMA registr.
5. Vyberte požadovanou přenosovou rychlost použitím USART_BRR registru.
6. Nastavte RE bit v registru USART_CR1. To povolí přijímač, který začne vyhledávat start bit.

Jakmile je přijat znak:

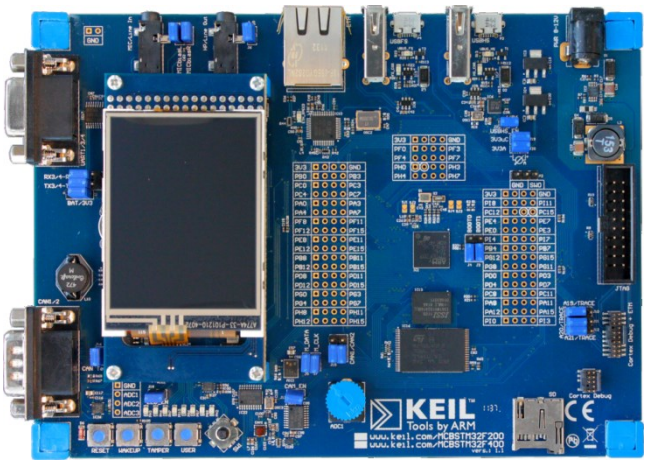
- RXNE bit je nastaven. Toto indikuje, že obsah shift registru přenesen do RDR. Jinak řečeno, data byla přijata a mohou být přečtena (stejně jako jejich přidružené error flagy).
- Přerušení je generováno, pokud je nastaven RXNE bit.
- Chybový příznak může být nastaven, pokud chybový rámec, šum nebo chyba přetečení byla detekována během příjmu.
- V multibufferu, RXNE je nastaven po každém přijatém bytu a je vyčištěn DMA čtením do datového registru.

- V single buffer módu je vyčištění RXNE bitu provedeno softwarovým čtením do registru USART_DR. RXNE příznak může být také vyčištěn, pokud od něj zapíšeme 0. RXNE bit musí být vyčištěn před koncem příjmu dalšího znaku pro vyhnutí se chyby přetečení.

4 Vývojové kity dostupné na katedře 352

Katedra 352 disponuje třemi vývojovými kity:

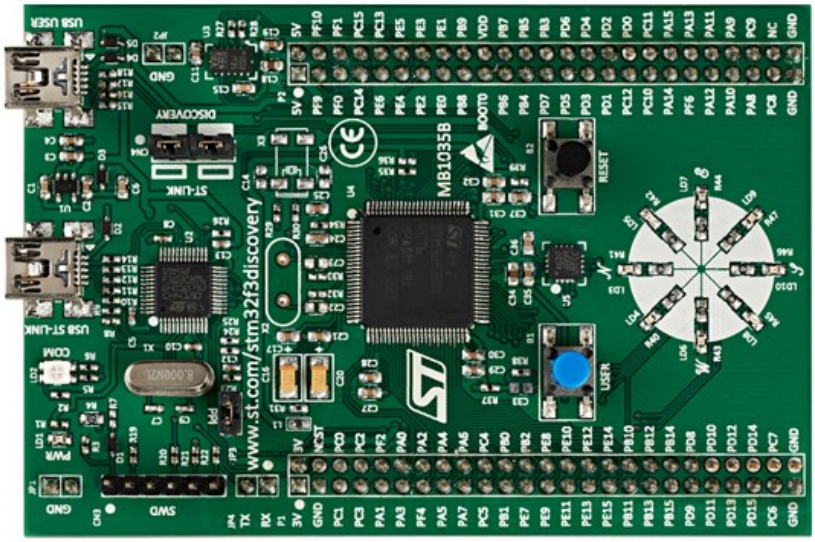
MCBSTM32F400



Obr. 23 Vývojový kit MCBSTM32F400[STMicroelectronics]

Tento kit disponuje procesorem STM32F407IG, který patří do řady s velmi vysokým výkonem. Na desce nalezneme také rozhraní pro CAN, UART, Ethernet, USB, microUSB a SD kartu. Kit je také vybaven 2,4 palcovým, barevným LCD dotykovým displejem a joystikem s pěti pozicemi. Dále je deska vybavena 1MB Flash a 192KB RAM pamětí a externími paměťmi 8MB NOR Flash, 512MB NAND Flash, 2MB SRAM, 8KB I^2C EEPROM s NFC rozhraním. K dispozici je také gyroskop, akcelerometr, digitální kamera a mikrofon, čtyři tlačítka, potenciometr a 8 LED diod. Malou nevýhodou této desky je, že nemá vestavěný nástroj pro debugování. Adaptér pro debugování je nutné připojit externě.

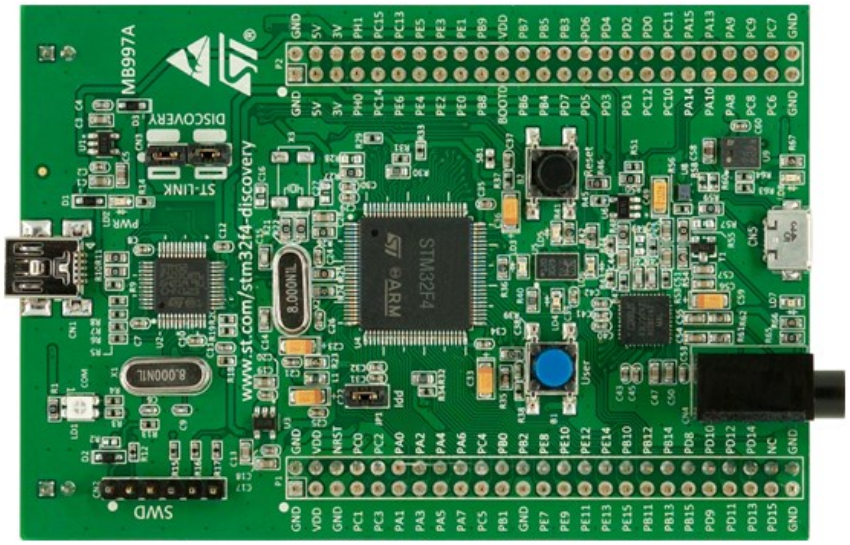
STM32F3-Discovery



Obr. 24 Vývojový kit STM32F3[STMicroelectronics]

Tato vývojová deska je osazena procesorem STM32F303VCT6 a je vybavena gyroskopem, akcelerometrem, MEMS pohybový sensor a e-compas, USB, deseti LED diodami a dvěma tlačítky. Výhodou oproti MCBSTM32F400 je, že má již vestavěný nástroj pro debugování ST-LINK/V2. Kit je vybaven 256 KB Flash pamětí a 48 KB RAM. Deska nepodporuje mikrokontroler STM32F313xx.

STM32F4-Discovery



Obr. 25 Vývojový kit STM32F4-Discovery[STMicroelectronics]

Na této desce najdeme procesor STM32F407VGT6, dva ST MEMS, digitální akcelerometr, digitální mikrofon, jeden audio digitální analogový převodník s integrovaným ovladačem třídy D, osm LED diod, dvě tlačítka a rozhraní pro microUSB. Dále je deska vybavena 1 MB Flash pamětí a 192 KB RAM. Na tomto kitu je také integrován nástroj pro debugování ST-LINK/V2.

5 Programovací prostředí Keil µVision 5

µVision 5 je vývojové prostředí vytvořené firmou Keil. V tomto prostředí je možné zakládat, spravovat, programovat nebo jinak editovat námi vytvořené nebo již předpřipravené projekty. Toto prostředí obsahuje také simulátor a debugger, který se používá k ladění programu.

5.1 Licenční edice Keil µVision

Vývojové prostředí je k dispozici ve čtyřech edicích:

MDK-Lite

Jde o freeware edici, která je volně stažitelná ze stránek Keil. Hlavní nevýhodou této edice je limit velikosti kódu, se kterým je možno pracovat. Maximální možná velikost kódu je 32KB. Oproti vyšším edicím také neobsahuje ARM Compiler Qualification Kit, knihovny Middleware a podporu ze strany Keil.

MDK-Cortex-M

Nejnižší placená edice. Oproti edici MDK-Lite má tu výhodu, že prostředí není limitováno velikostí kódu a je zde poskytnuta podpora ze strany firmy Keil po dobu 12 měsíců. Tato edice také nepodporuje ARM7™, ARM9™, ARM® Cortex®-R4, ARM® SecurCore®.

MDK-Standart, nyní se mění na verzi PLUS

Standartní edice. Oproti edici MDK-Cortex-M obsahuje podporu ARM7™, ARM9™, ARM® Cortex®-R4, ARM® SecurCore®.

MDK-Professional

Nejvyšší edice prostředí Keil µVision. Oproti nižším verzím obsahuje ARM Compiler Qualification Kit a knihovny Middleware.

Licence se dále dělí na dva typy: Node-Locked a Floating.

Node-Locked licence umožňuje používání pouze jedné osobě na maximálně dvou počítačích. Pro tuto licenci není potřeba připojení počítače k síti.

Typ Floating-User umožní vývojářům používání softwaru na více počítačích. Tento typ licence vyžaduje souborový server, na kterém je hostován licenční soubor. Je zde také vyžadováno připojení k síti, pro kontrolu floating licence.

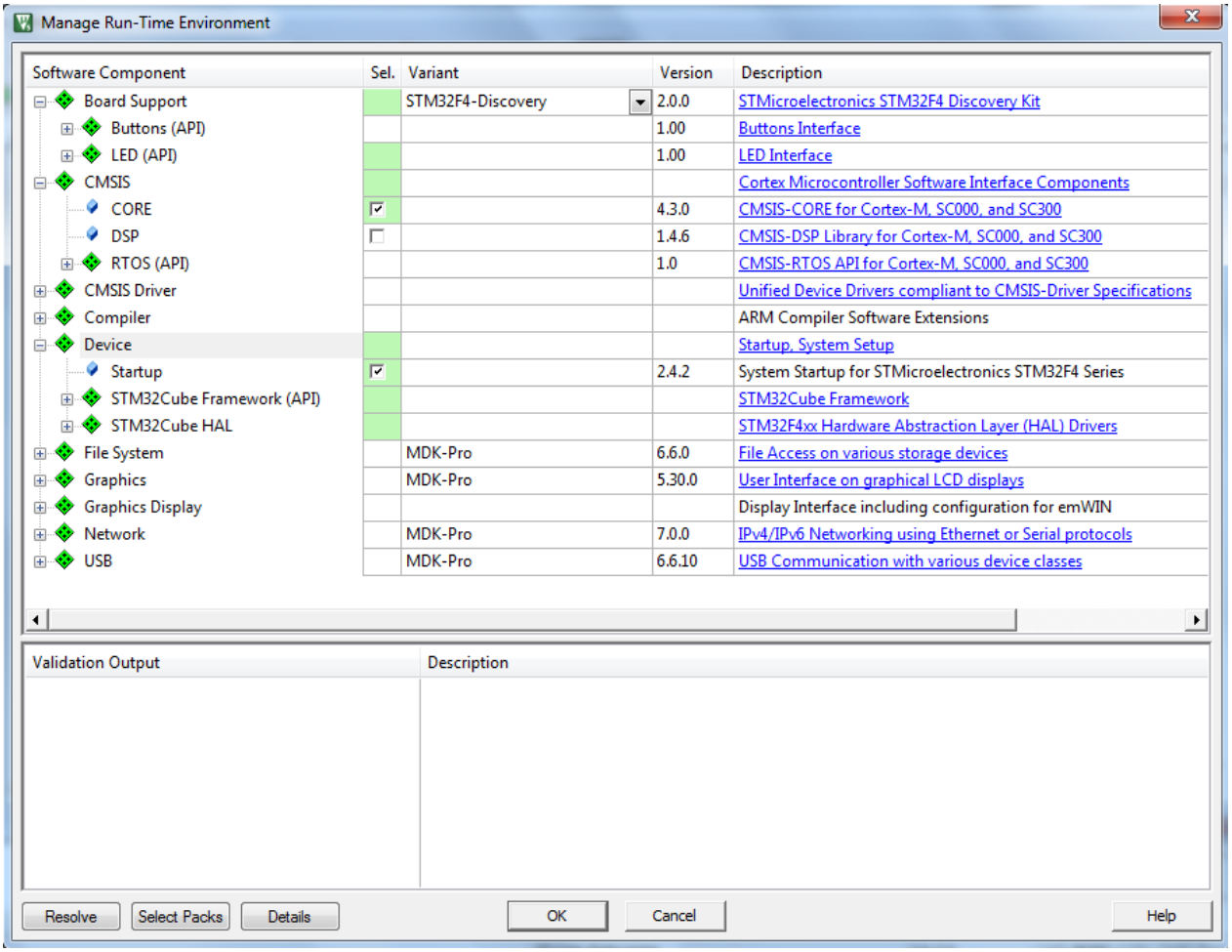
5.2 µVision

Aplikace µVision je software pro vývojáře, který kombinuje robustní a moderní editor s projekt manažerem a nástroj pro jednoduché sestavování. Integruje v sobě všechny nástroje pro vývoj aplikací, které zahrnují C/C++ kompilátor, makro assembler a generátor HEX souborů. µVision pomáhá urychlit vývojářský proces aplikací tím, že poskytuje:

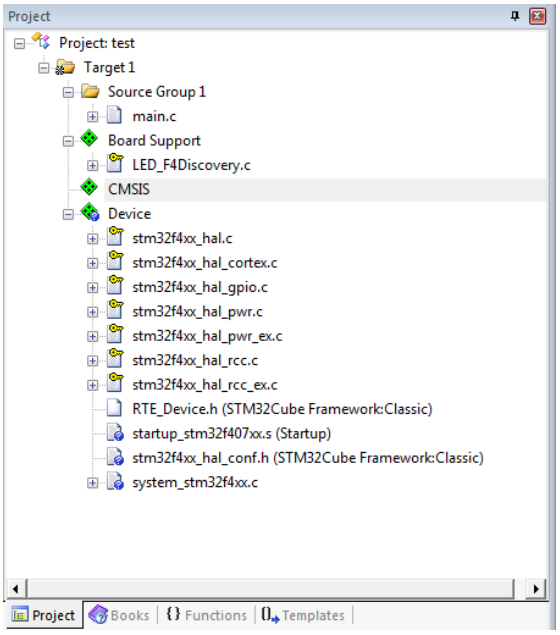
- Editor zdrojového kódu s mnoha podpůrnými funkcemi
- Databázi zařízení pro nastavení vývojového nástroje
- Projekt manažer pro vytváření a správu projektů
- Integrovanou funkci pro sestavení, kompilování a linkování aplikace
- Dialogy pro nastavení vývojářského prostředí
- Integrovaný debugger na úrovni zdrojového kódu a asembleru
- Simulátor periférií
- Utilitu pro nahrání aplikace do Flash paměti
- Odkazy na manuály, on-line nápovědu, datasheety zařízení a uživatelské příručky

µVision Projekt Manažer a Run-Time Environment

Pomocí Projekt Manažeru a Run-Time Environment můžeme vytvářet aplikace použitím již vytvořených softwarových komponent a podpory pro zařízení ze softwarových balíčků. Softwarové komponenty se skládají z knihoven, zdrojových modulů, konfiguračních souborů, šablon zdrojových souborů a dokumentace. Softwarové komponenty mohou být generické pro podporu širokého rozsahu zařízení a aplikací.



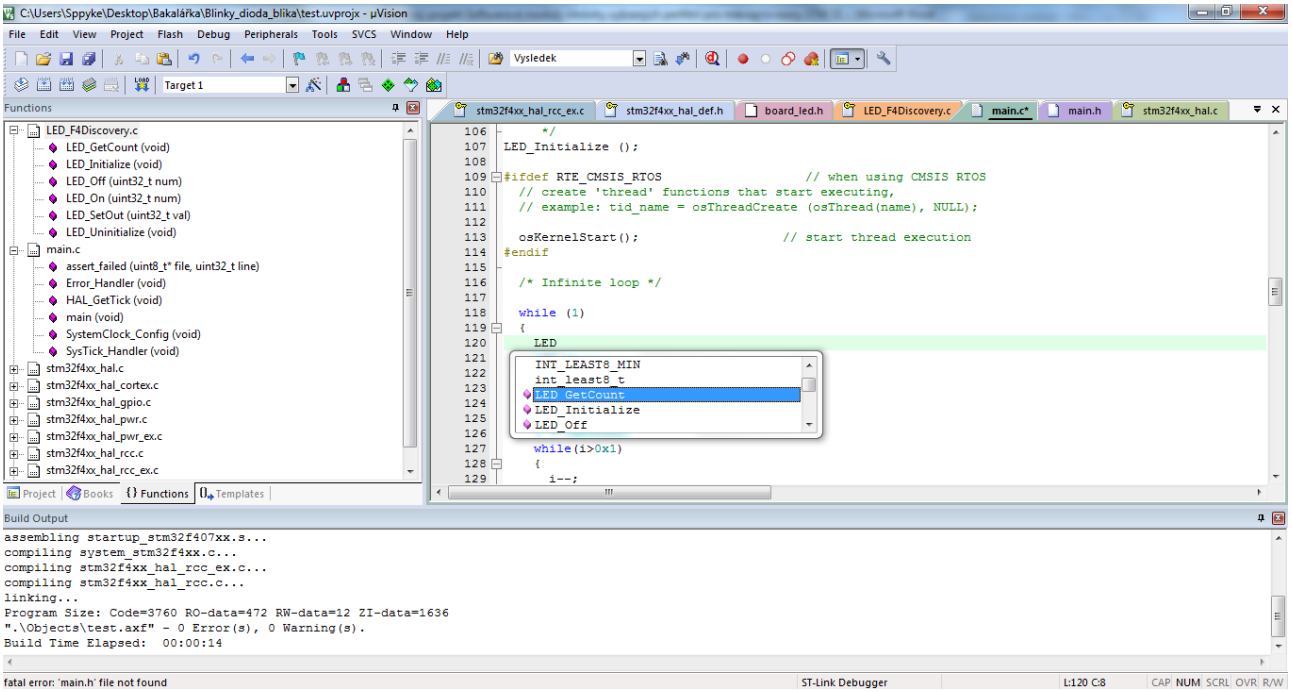
Obr. 26 OknoManage Run-Time Environment



Obr. 27 Projekt Manažer

µVision Editor

Integrovaný µVision Editor zahrnuje všechny standartní vlastnosti moderního editoru zdrojového kódu. Editor je dostupný i během debugování a barevná syntaxe, zvýrazňování, textové odsazení a našeptávání kódu je optimalizováno pro C/C++. Okno funkcí poskytuje rychlý přístup k funkcím v každém softwarovém modulu. Dynamická kontrola syntaxe ověřuje syntaxi programu během psaní kódu a poskytuje varování při porušení pravidel programování v reálném čase.



Obr. 28 µVision Editor

Průvodce instalací, nastavením a praktickou ukázkou práce s vývojovým prostředím KEIL µVision 5 je k nalezení příloze A.

6 Závěr

Hlavním cílem bakalářské práce bylo seznámit se s vývojovým prostředím KEIL a s podpůrnou knihovnou STM32Cube. Dalším důležitým bodem zadání bylo vytvořit studijní opory vytvořených příkladů. Tyto studijní opory se věnují především práci se standartním vstupem a výstupem a ovládání LCD dotykového displeje. Knihovna STM32Cube je zde zastoupena v podobě widgetů, které jsou zobrazovány na displeji.

V první polovině teoretické části práce byl zpracován popis mikroprocesorů řady STM32. Řada se dále dělí na tři série. První z nich je řada: L0, L1, L4. Tato řada klade důraz především na nízkou spotřebu energie, proto je nejvhodnější do zařízení, která jsou napájena baterií nebo tam, kde se energie získává z vnějších zdrojů. Druhou řadu tvoří: F0, F1, F3. Jde o hlavní tzv. mainstreamovou řadu, která klade důraz v první řadě na cenu. Proto se dbá na dobrý poměr výkonu a spotřeby. Tyto produkty jsou nejvhodnější tam, kde cena je jedním z hlavních parametrů zařízení. Třetí a zároveň nejvyšší řadou produktů STM32 je řada: F2, F4, F7. Tato série byla navržena s maximálním důrazem na výkon mikroprocesorů a v několika nezávislých testech výkonu mikroprocesorů získaly produkty této série nejvyšší bodové hodnocení. Druhá část teoretické části se zabývá popisem a základním ovládáním vývojového prostředí KEIL µVision 5. Je zde také zpracován popis a možnosti jednotlivých licenčních edicí tohoto prostředí.

Praktická část práce se věnuje vytvoření sady jednoduchých úloh, které byly převedeny do podoby studijních materiálů. První úloha slouží k demonstraci fungování vývojového prostředí KEIL µVision 5. Na příkladu předpřipraveného projektu pro vývojový kit STM32F4-Discovery je vysvětleno ovládání a základní fungování prostředí. Druhý příklad se věnuje založení projektu a přidání potřebných souborů pomocí okna Manage Run-Time Environment. V úloze je dále podrobně popsáno jak program upravit tak, aby na desce STM32F4 – Discovery začala blikat dioda. Ve třetí úloze je podrobně popsán postup vytvoření kostry programu pro práci s deskou STM32F4 – Discovery spolu s rozšiřující deskou STM32F4DISCOVERY Base Board a LCD dotykovým displejem 3.5" LCD board. Do programu jsou poté přidána celkem tři okna. První okno slouží jako navigace do dalších oken programu. Je zde vysvětleno fungování aplikace, která řídí rozsvícení LED diody na kitu. Úloha také obsahuje popis a vysvětlení fungování aplikace pro jednoduché výpočty ze dvou uživatelem zadaných hodnot.

Třetí úlohu by bylo možné dále rozšířit o další aplikace, které by blíže vysvětlily a popsaly možnosti knihovny STM32Cube. Aplikace by mohly obsahovat větší množství widgetů, které tato knihovna nabízí. Dále by bylo by možné také zpracovat podrobný popis zdrojových souborů, které konfigurují LCD displej a také jeho komunikaci s deskou STM32F4 – Discovery.

7 Použitá literatura

ARM KEIL. *Getting StartedCreate Applications with MDK Version 5for ARM®Cortex®-M Microcontrollers*. ARM Germany HmbH, 2015 dostupné z <URL: http://www2.keil.com/docs/default-source/default-document-library/mdk5-getting-started.pdf>

FARANA, Radim, Lubomír SMUTNÝ a Antonín VÍTEČEK. *Zpracování odborných textů z oblasti automatizace a informatiky*. 1. vyd. Ostrava: VŠB-Technická univerzita, 1999. ISBN 80-7078-737-6.

Sasang Balachandran. *General Purpose Input/Output (GPIO)* [online]. 2009 [cit. 2016-04-10]. Dostupné z: http://www.egr.msu.edu/classes/ece480/capstone/fall09/group03/AN_balachandran.pdf

HEROUT, Pavel. *Učebnice jazyka C*. 4., přeprac. vyd. České Budějovice: Kopp, 2004, 271, viii s. ISBN 80-7232-220-6.

HRBÁČEK, Jiří. *Moderní učebnice programování jednočipových mikrokontrolérů PIC*. Praha: BEN - technická literatura, 2004. ISBN 80-7300-136-5.

MATOUŠEK, David. *C pro mikrokontoléry [sic] ATMEL AT89S52: příklady a aplikace pro C51 ve vývojovém prostředí KEIL [mý]Vision3* [CD-ROM]. Praha: BEN - technická literatura, 2007. ISBN 978-80-7300-215-2.

SEAL, David (ed.). *ARM architecture reference manual*. 2nd ed. Harlow: Addison-Wesley, 2001. ISBN 0-201-73719-1.

STMicroelectronics. Microcontrollers [online]. 2016 [cit. 2016-01-12]. Dostupné z: http://www.st.com/web/en/catalog/mmc/FM141

STMicroelectronics 2015.*RM0090 Reference manual STM32F405/415, STM32F407/417, STM32F427/437 and STM32F429/439 advanced ARM®-based 32-bit MCUs* [online]. 2015 [cit. 2016-04-19]. Dostupné z: http://www2.st.com/content/ccc/resource/technical/document/reference_manual/3d/6d/5a/66/b4/99/40/d4/DM00031020.pdf/files/DM00031020.pdf/jcr:content/translations/en.DM00031020.pdf

VÁŇA, Vladimír. *ARM pro začátečníky*. 1. vyd. Praha: BEN - technická literatura, 2009, 195 s. ISBN 978-80-7300-246-6.

Přílohy

Příloha A: Seznámení se s vývojovým prostředím KEIL µVision 5

Po úspěšném a aktivním absolvování této KAPITOLY

Budete umět:	Budete umět
• Založit projekt v prostředí KEIL µVision 5 • Stáhnout a spustit předpřipravený projekt	

Budete schopni:	Budete schopni
• Provádět základní operace v prostředí KEIL µVision 5 • Upravovat projekt v prostředí KEIL µVision 5	

Tato kapitola si klade za cíl seznámit uživatele s vývojovým prostředím KEIL µVision 5. Bude zde rozebráno k čemu je toto prostředí určeno, jaké jsou jeho základní vlastnosti a funkce, kde můžeme toto prostředí získat. Dále v kapitole je popsáno jak pomocí tohoto vývojového prostředí stáhnout již hotový ukázkový program a jak v něm provádět drobné úpravy.

Čas ke studiu: 2 hodiny

Cíl Po prostudování tohoto kapitoly budete umět

- Provádět základní operace ve vývojovém prostředí KEIL µVision 5
- Založit projekt v prostředí KEIL µVision 5
- Stáhnout a spustit již připravený projekt
- Provádět úpravy v projektu

Výklad

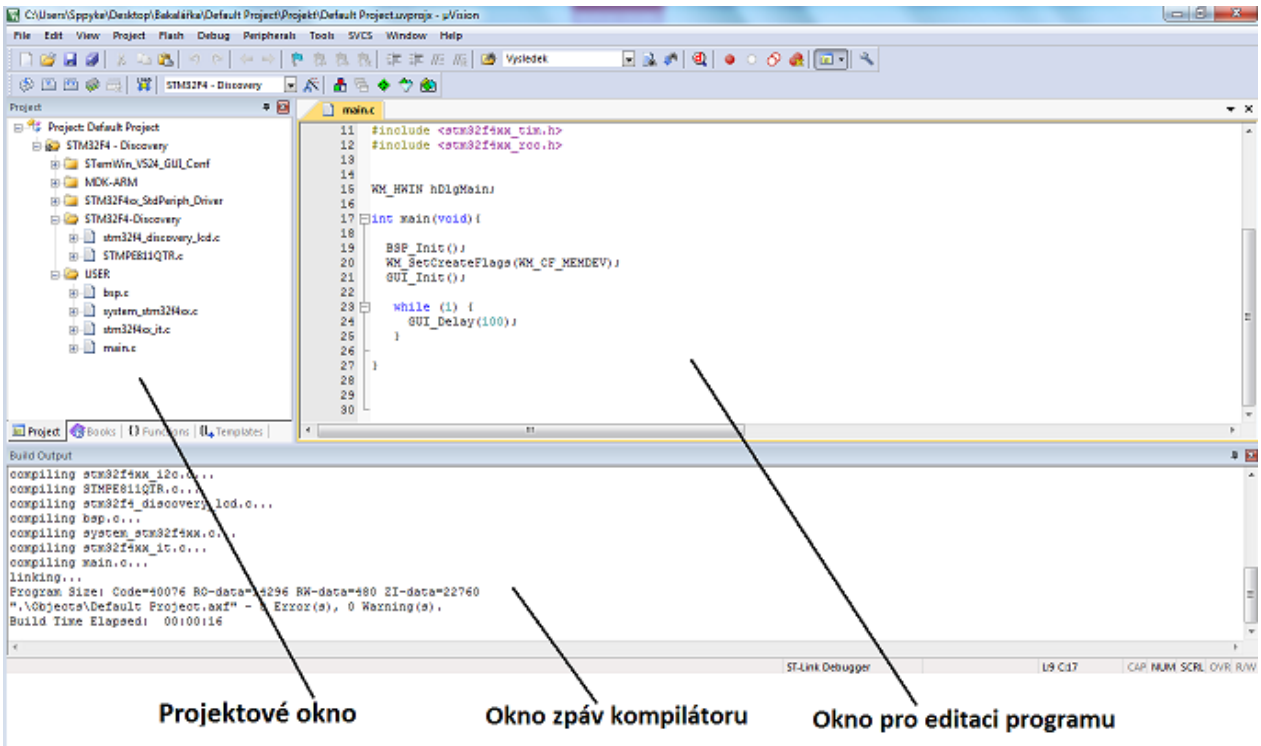
A.1 KEIL µVision 5

KEIL µVision 5 je vývojové prostředí vyvíjené firmou KEIL. Je to program, který je určen pro vytváření, správu a ladění projektů. Ve velké míře se používá pro vývoj programů pro mikrokontrolery téměř všech značek a typů. Jde o velmi moderní a přehledné prostředí, ve kterém se velmi snadno orientuje a pracuje.

A.1.1 Stažení a instalace prostředí KEIL µVision 5

Vývojové prostředí KEIL µVision 5 je k dispozici volně ke stažení ze stránek společnosti KEIL (odkaz pro stažení: https://www.keil.com/demo/eval/arm.htm). Před samotným stažením je potřeba vyplnit formulář s důležitými údaji pro společnost KEIL. Po vyplnění těchto údajů již budeme mít v prohlížeči k dispozici odkaz na stažení instalačního souboru s příponou exe. Po stažení spustíme soubor a dále postupujeme podle instrukcí v instalačním programu. Při prvním otevření vývojového studia se nám také otevře okno Pack Installer. V tomto okně bude potřeba aktualizovat zastaralé soubory a stáhnout balíček souborů pro mikrokontroler, který budeme chtít programovat. V našem případě se soustředíme na desku STM32F4-Discovery. Proto v okně Pack Installer v levé části okna klepneme na záložku boards a najdeme desku STM32F4-Discovery. V pravé části okna Pack Installer klepneme na záložku Packs a poté nainstalujeme komponenty: Keil::STM32F4xx_DFP, ARM::CMSIS, Keil::ARM_Compiler a Keil::MDK-Middleware. Nyní by prostředí mělo být připraveno k použití.

A.1.2 Seznámení se s protředím KEIL µVision 5



Obr. 29 Vývojové prostředí KEIL µVision 5

Na výše uvedeném obrázku je vyobrazeno prostředí µVision 5 při správě otevřeného projektu. Prostředí můžeme rozdělit do tří hlavních částí. První částí je projektové okno. V tomto okně vidíme všechny knihovny, zdrojové a hlavičkové soubory, spouštěcí soubory, které jsou do projektu přidány. Pro přehlednost projektu je velmi vhodné tyto soubory rozdělit do složek, aby se při úpravách daly tyto soubory snadno najít. Druhou částí je okno zpráv kompilátoru. V tomto okně

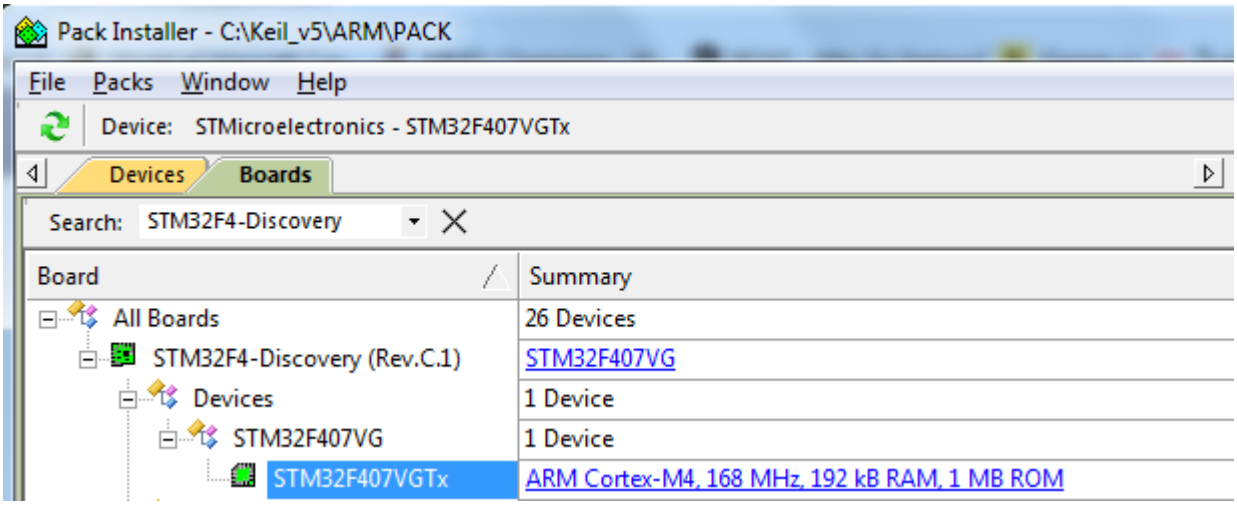
jsou zobrazovány důležité informace, které nám chce kompilátor předat. Jde o informace typu, zda byl projekt zkompilován bez chyb, zda při linkování nastala nějaká chyba atp. V případě, že se kód v projektu nepodařilo sestavit, jsou v tomto okně k dispozici informace pro nalezení chyby v kódu, která brání sestavení projektu. Po kompilaci programu je zde informace o tom, zda byla kompilace provedena úspěšně. Pokud nebyla, jsou informace o počtu a umístění chyb, dále je zde počet varování, které nám chce kompilátor sdělit. Třetí částí je okno pro editaci programu. Toto okno slouží především pro psaní kódu, který chceme aby mikrokontroler vykonal. Najdeme zde také informaci o tom, na kterém řádku se nacházíme, a je zde také možnost vložení breakpointu. V horní části vývojového prostředí nalezneme lištu s nabídkou pro editaci projektu, uživatelské nastavení prostředí, nastavení periférií a debugování, nastavení oken ve vývojovém prostředí a v neposlední řadě zde nalezneme také nápovědu. Pod horní lištou nalezneme také nemalé množství ikon, které slouží jako rychlý přístup k velmi používaným volbám v horní liště. Jsou zde ikony například pro uložení projektu, přidání komentáře do kódu, spuštění debuggeru, spuštění kompilace atp.

A.1.3 Předpřipravený program

K bližšímu seznámení s prostředím nám pomůže spuštění programu, který je volně stažitelný ze stránek firmy Keil. Stažení samotné lze provést i v prostředí µVision pomocí tzv. Pack Installeru. Tato utilita slouží ke stahování různých balíčků pro vývojové kity nebo zařízení, které chceme v našich projektech používat.

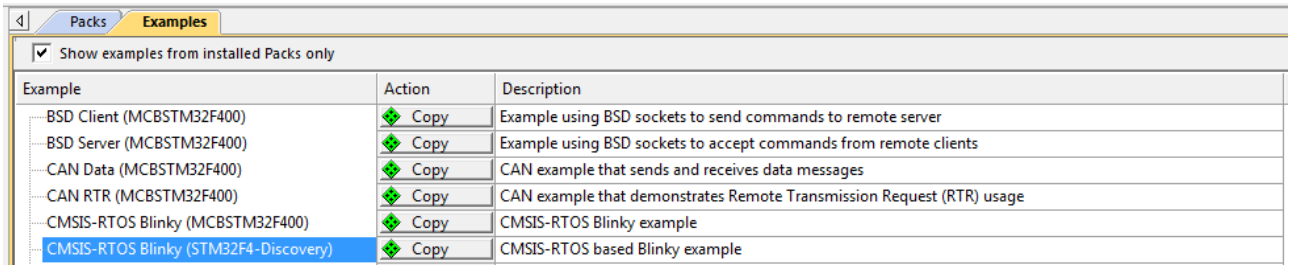
Obr. 30 Ikona Pack Installeru v prostředí Keil µVision 5

Nyní je nutné v levé části obrazovky najít zařízení, ke kterému chceme najít předpřipravený příklad. V našem případě jde o vývojový kit STM32F4-Discovery od firmy STMicroelectronics. Klikneme proto na záložku Boards a najdeme náš kit. Můžeme využít vyhledávací okno pro rychlejší nalezení potřebného zařízení.



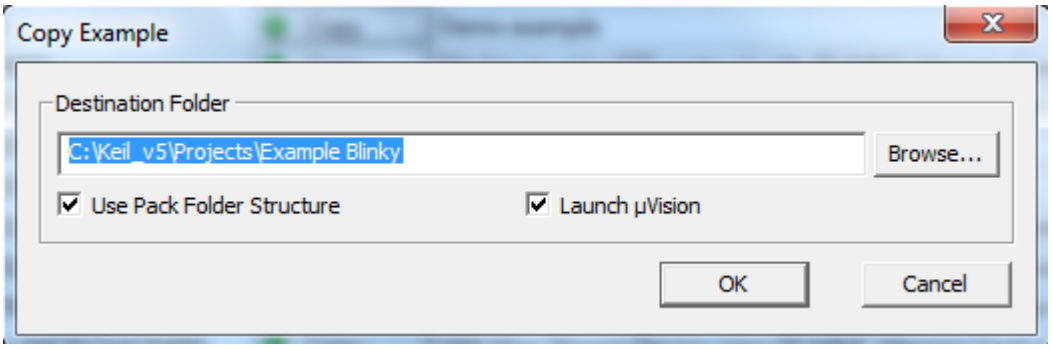
Obr. 31 Výběr použitého kitu

V pravé části okna máme nyní k dispozici balíčky a příklady, které jsou kompatibilní s námi vybraným kitem. Klikneme proto na záložku Examples a poté na tlačítko Copy u příkladu CMSIS-RTOS Blinky (STM32F4-Discovery).



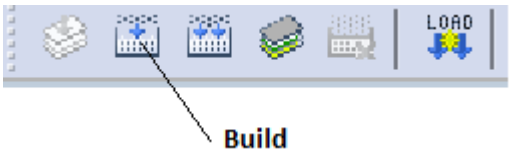
Obr. 32 Výběr příkladu v okně Pack Installeru

Prostředí se nás v tomto kroku zeptá, do které složky chceme příklad zkopírovat. Proto klikneme na tlačítko Browse a vytvoříme si složku v místě, kde chceme program uložit.

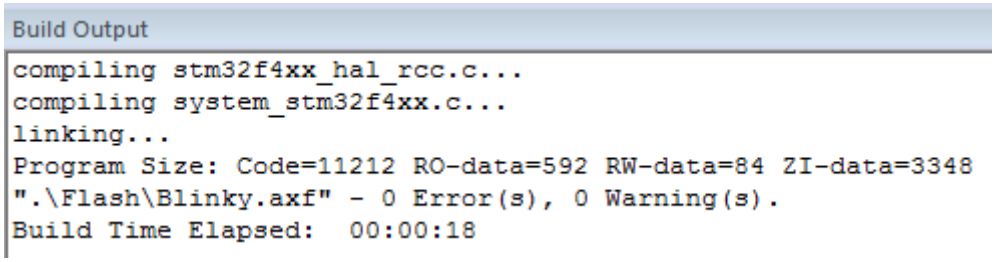


Obr. 33 Nastavení cesty do složky, kde bude program zkopírován

Nyní program zkompilujeme pomocí tlačítko Build. V oznamovací části by prostředí nemělo hlásit žádné chyby ani upozornění. Pokud se tak nestane, někde jsme udělali chybu.



Obr. 34 Tlačítko Build v prostředí Keil µVision 5



Obr. 35 Oznamovací okno v prostředí Keil µVision 5

Před samotným nahráním programu do paměti musíme ještě nastavit debugger. Klikneme proto na tlačítko Options for Target…Pak klepneme na záložku Debug a nastavíme debugger na ST-Link Debugger. V nastavení debuggeru ještě změníme port na SW a jsme hotovi.

Obr. 36 Tlačítko Options for Target v prostředí Keil µVision 5

Nyní jsme připraveni připojit náš kit pomocí USB do počítače a pomocí tlačítka download nahrát program do Flash paměti vývojového kitu.

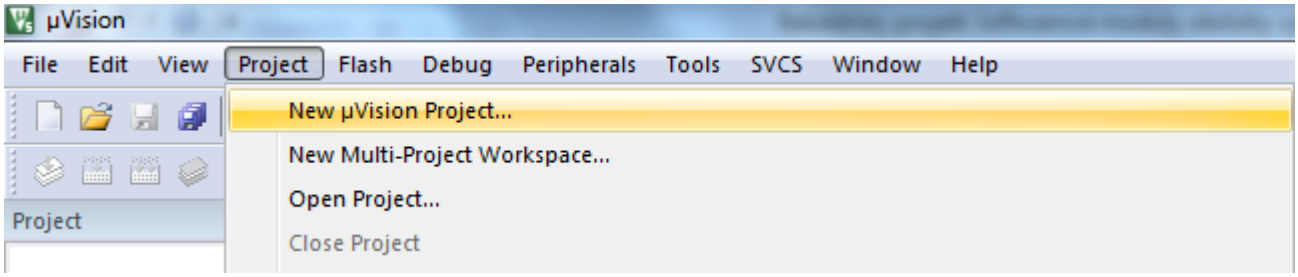
Obr. 37 Tlačítko download v prostředí Keil µVision 5

Program by měl nyní správně fungovat. Diody na kitu by měly blikat dokola. Blikání je možno zastavit držením tlačítka User. Po jeho uvolnění bude blikání opět pokračovat.

A.2 Projekt Blinky

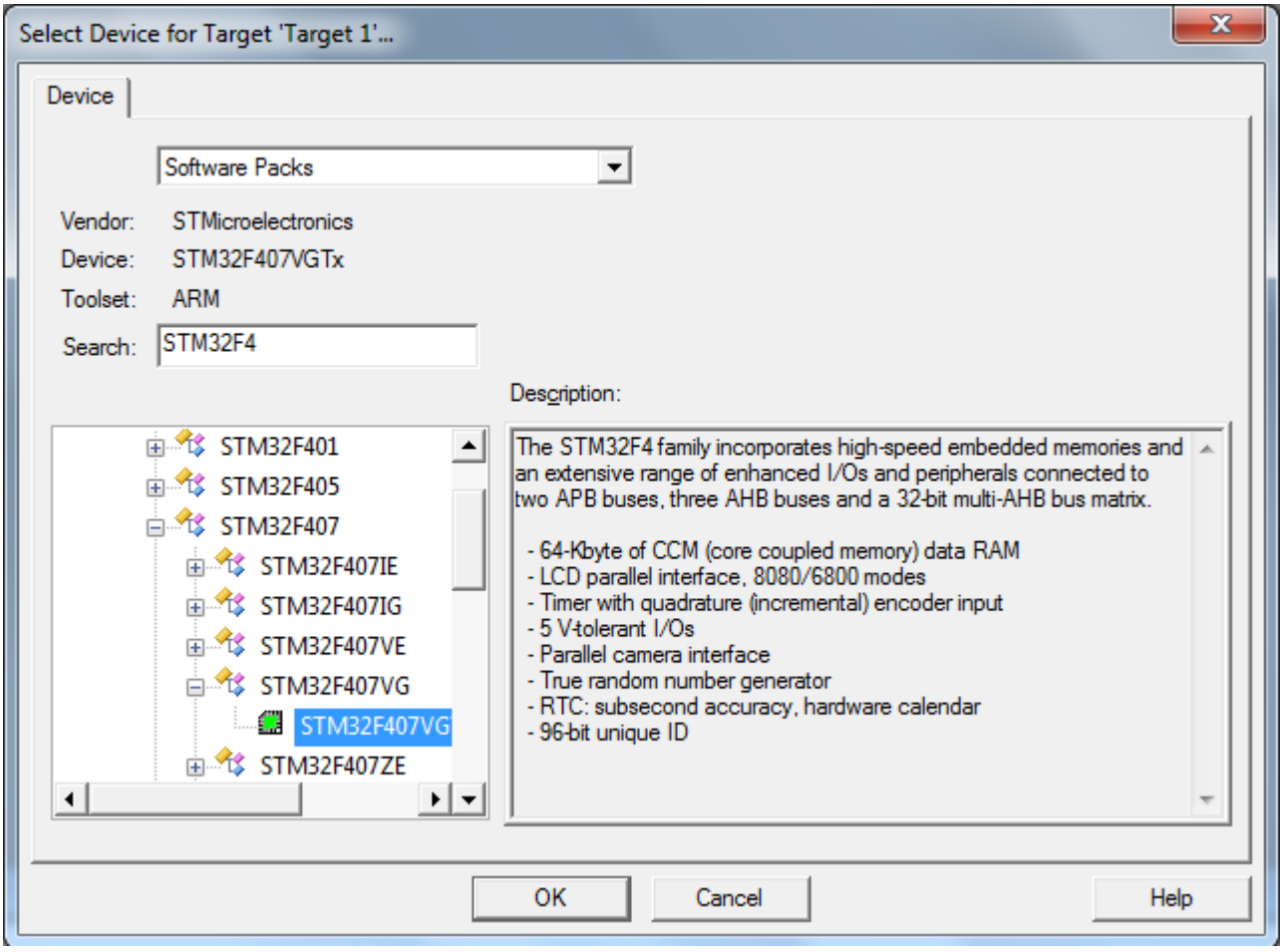
V tomto projektu si předvedeme jak vytvořit projekt s blikající diodou úplně od začátku. Vytvoříme si projekt ve vývojovém prostředí, přidáme potřebné zdrojové soubory a nakonec upravíme funkci main() tak, aby na našem zařízení blikala dioda, kterou si určíme. Pro tento náš projekt použijeme opět vývojový kit STM32F4-Discovery.

První věc, kterou je třeba udělat je vytvořit si nový projekt v prostředí µVision. Projekt založíme klepnutím na možnost New µVision Project… v záložce Project.



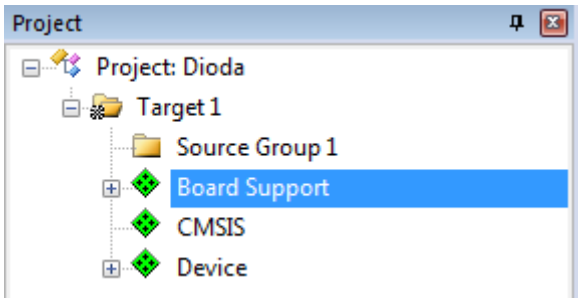
Obr. 38 Založení nového projektu v prostředí Keil µVision 5

Program je v této chvíli nutné pojmenovat a uložit do složky, kde potřebujeme. Poté klikneme na tlačítko uložit a projekt bude vytvořen. Po vytvoření projektu je nutné vybrat zařízení, se kterým budeme v projektu pracovat. V nabídce proto najdeme zařízení STM32F407VG, které je osazeno na desce STM32F4-Discovery. Po vybrání zařízení klepneme na tlačítko OK a vyskočí okno Manage Run-Time Environment. Toto okno slouží k výběru komponent, které budeme v programu potřebovat.



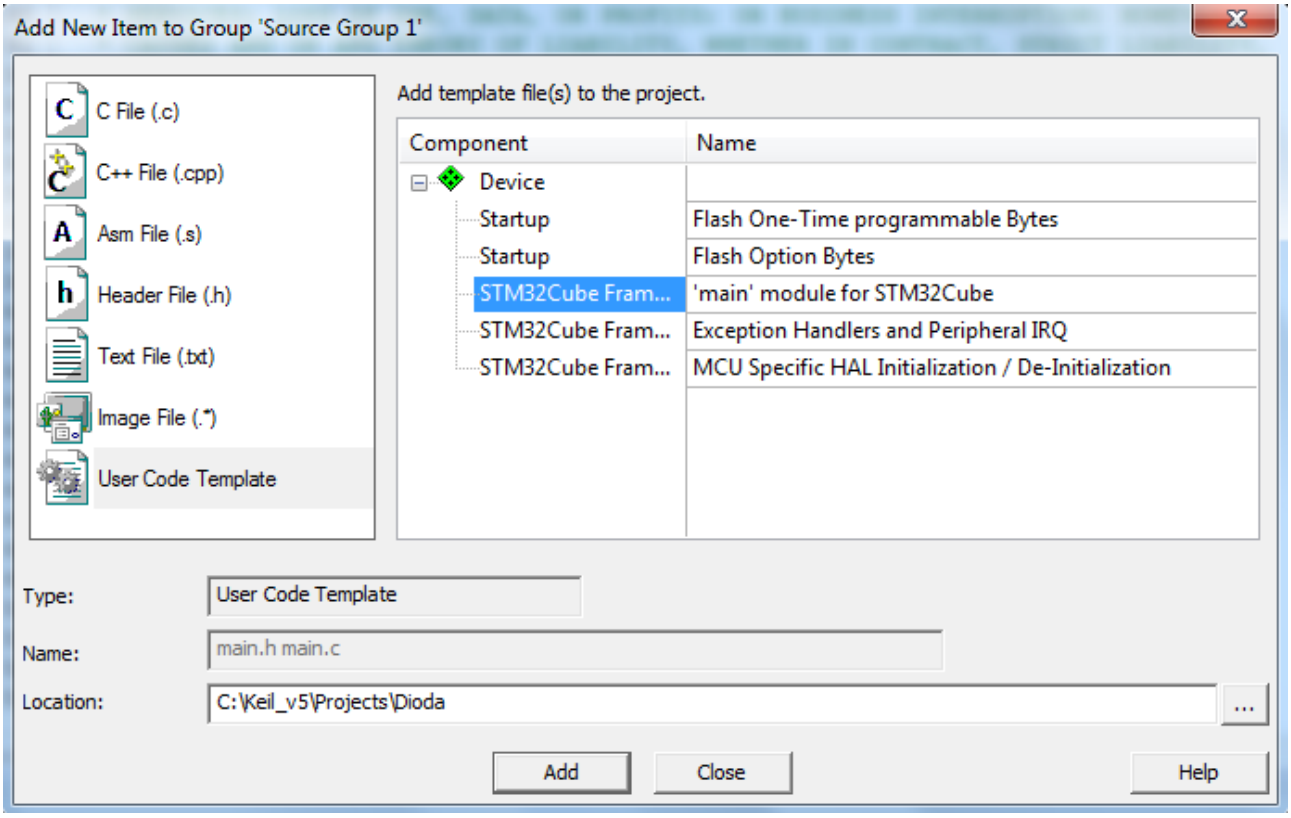
Obr. 39 Výběr zařízení, které bude použito v projektu

V okně Manage Run-Time Environment nastavíme podporu desky STM32F4-Discovery a rozbalíme záložky Board Support a LED (API) a zaškrtneme možnost LED. Dále rozbalíme záložku CMSIS a zaškrtneme možnost CORE. Nyní v záložce Device zaškrtneme možnost Startup a dále v záložce STM32Cube Framework (API) vybereme možnost Classic. Nakonec v záložce STM32Cube HAL zaškrtneme možnosti Common, Cortex, GPIO, PWR a RCC. Pokud v oznamovací části okna není žádné chybové hlášení potvrdíme výběr stiskem tlačítka OK. Nyní můžeme rozbalit v okně Projekt možnost Target 1, kde uvidíme naše zařízení spolu s komponenty, které jsme vybrali.



Obr. 40 Okno projektu po přidání komponent v projektu Blinky pro desku STM32F4-Discovery

Nyní je nutné přidat funkci main. Klikneme pravým tlačítkem na složku Source Group 1 a poté na možnost Add New Item to Group. Vybereme možnost User Code Template, rozbalíme záložku Device a klikneme na main funkci a potvrdíme.



Obr. 41 Přidání funkce main do projektu

Main funkce je nyní vložena. Teď do těla funkce dopíšeme kód, který rozbliká diodu. Nejdříve však musíme zahrnou do funkce main i knihovnu board_led.h. Toho docílíme napsáním příkazu #include "board_led.h" na začátku souboru. Dále je třeba definovat funkci SysTick_Handler. Proto za funkci main napíšeme definici funkce SysTick_Handler. Do funkce main poté napíšeme příkaz LED_Initialize(); pro inicializaci diod. Nakonec napíšeme do nekonečné smyčky na konci funkce main program, který rozsvítí diodu, poté chvíli počká, vypne diodu a zase počká stejnou chvíli.

```
/* Infinite loop */

while (1)
{
  LED_On(1);
  i = 0xFFFFFF;
  while(i>0x1)
    { i--;}
  LED_Off(1);
  i = 0xFFFFFF;
  while(i>0x1)
  {
    i--;
  }
}
```

Obr. 43 Program v nekonečné smyčce

```
int f = 0;
void SysTick_Handler(void)
{
   f++;
}
```

Obr. 42 Definice funkce SysTick_Handler

Před samotným zkompilováním programu a jeho nahráním do paměti musíme ještě nastavit debugger. Klikneme proto na tlačítko Options for Target…Pak klepneme na záložku Debug a nastavíme debugger na ST-Link Debugger. V nastavení debuggeru ještě změníme port na SW a jsme hotovi. Po připojení desky k počítači pomocí USB můžeme program sestavit a nahrát do paměti. Pokud jsme vše udělali správně dioda na desce bude blikat.

Úkol k řešení

1. Změňte frekvenci blikání diod v předpřipraveném příkladu.
2. Změňte program v projektu Blinky tak, aby diody blikaly dokola.

Shrnutí kapitoly

V této kapitole jsme se naučili provádět základní úkony v prostředí KEIL µVision 5. Umíme se v prostředí orientovat, umíme používat Pack Installer, a naučili jsme se provádět drobné úpravy v programu.

V druhé části kapitoly jsme se naučili jak vytvořit projekt v prostředí KEIL µVision 5. Dále jsme se naučili vytvořit jednoduchý program, který rozbliká diodu na desce STM32F4-Discovery.

Klíčová slova

STM32F4-Discovery, Blinky, KEIL, programování

Kontrolní otázka 1

K jakému účelu je primárně určeno vývojové prostředí KEIL µVision 5?

Prostředí KEIL µVision 5 je primárně určeno k vytváření programů pro mikrokontrolery.

Kontrolní otázka 2

Jaké typy mikrokontrolerů podporuje prostředí KEIL µVision 5?

Prostředí KEIL µVision 5 podporuje téměř všechny typy mikrokontrolerů

Klíč k řešení

1. V souboru Blinky.c je na řádku č. 89 funkce čekání osDelay(). Tato funkce se stará o frekvenci blikání diod. Pokud číslo v parametru funkce snížíme, budou diody blikat rychleji. V opačném případě bude prodleva mezi blikáním větší.

2. Na níže uvedeném obrázku vidíte úpravu programu v nekonečné smyčce ve funkci main().

```
/* Infinite loop */

while (1)
{
  for (j=1;j<5;j++)
  {
  LED_On(j);
  i = 0xFFFFFF;
  while(i>0x1)
    { i--;}
  LED_Off(j);
  i = 0xFFFFFF;
  while(i>0x1)
  {
    i--;
  }
  }

}
```

Obr. 44 Upravený program v nekonečné smyčce

Příloha B: Projekt pro desku STM32F4 – Discovery s podporou knihovny STM32Cube

Po úspěšném a aktivním absolvování této KAPITOLY

Budete umět:	Budete umět
• Připravit projekt pro desku STM32F4-Discovery s připojeným LCD dotykovým displejem s podporou knihovny STM32Cube • Vytvořit program pro ovládání LED diody na desce STM32F4-Discovery pomocí rozhraní na displeji • Vytvořit program, který bude fungovat jako kalkulátor pomocí grafického rozhraní na LCD displeji	

Budete schopni:	Budete schopni
• Ovládat aplikaci STM32Cube pomocí oken a tlačítek • Ovládat diodu na desce STM32F4-Discovery pomocí grafického rozhraní • Používat utilitu GUIBuilder	

Tato kapitola si klade za cíl naučit uživatele pracovat s knihovnou STM32Cube pomocí jednoduchých příkladů. Kapitole se věnuje především prvkům, jako jsou okna a tlačítka, což jsou základní prvky, které jsou potřeba v každé aplikaci, která používá grafické rozhraní. Začátek kapitoly je věnován stručnému popisu knihovny STM32Cube. Dále je zde popsán projekt, pomocí kterého je vytvořeno grafické rozhraní ovládání diody na desce STM32Cube. Poslední část kapitoly se věnuje projektu Calculator. Tento projekt funguje jako kalkulátor, který je vytvořen pomocí prvků, které nabízí knihovna STM32.

Čas ke studiu: 3 hodiny

Cíl Po prostudování tohoto kapitoly budete umět

- Vytvořit základní projekt pro desku STM32F4 - Discovery
- Vytvořit aplikaci pro ovládání diody pomocí grafického rozhraní
- Vytvořit aplikaci kalkulačky, pomocí prvků knihovny STM32Cube

Výklad

B.1 STM32Cube

STM32Cube je software vyvinutý firmou STMicroelectronics, který se snaží usnadnit práci vývojářům snížením času a také peněz potřebného k vývoji jejich projektu.

STM32Cube se skládá ze dvou částí: první z nich je STM32CubeMX, což je program, který umožňuje generovat kód v jazyce C pomocí grafického rozhraní. Pomocí tohoto programu je možné založit projekt a vytvořit základní kostru programu. Tento projekt je možné dále otevřít v prostředí Keil µVision 5 a dále jej editovat.

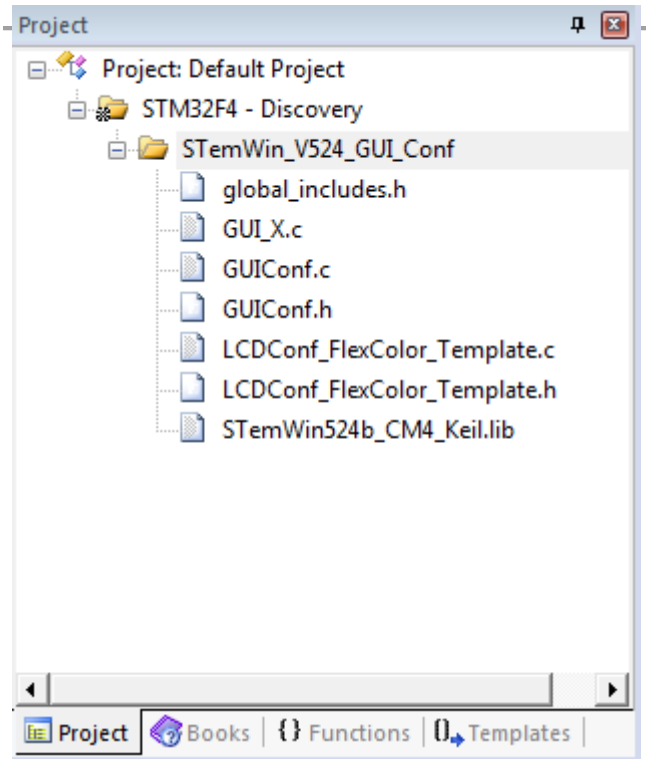
Druhou částí balíčku STM32Cube jsou knihovny, které obsahují například HAL vrstvu, která umožňuje přenosnost mezi zařízeními STM32. Knihovny také obsahují sbírku middleware komponent jako RTOS, USB knihovnu, souborový systém, zásobník TCP/IP, dotykové snímání nebo grafickou knihovnu. Každá komponenta má svůj jednoduchý projekt, kde je vysvětleno jak tuto knihovnu inicializovat a jak jí používat. Každý tento projekt je možné editovat a tak je možné jej použít jako základ při vytváření projektu dle vlastních potřeb.

B.2 Příprava projektu

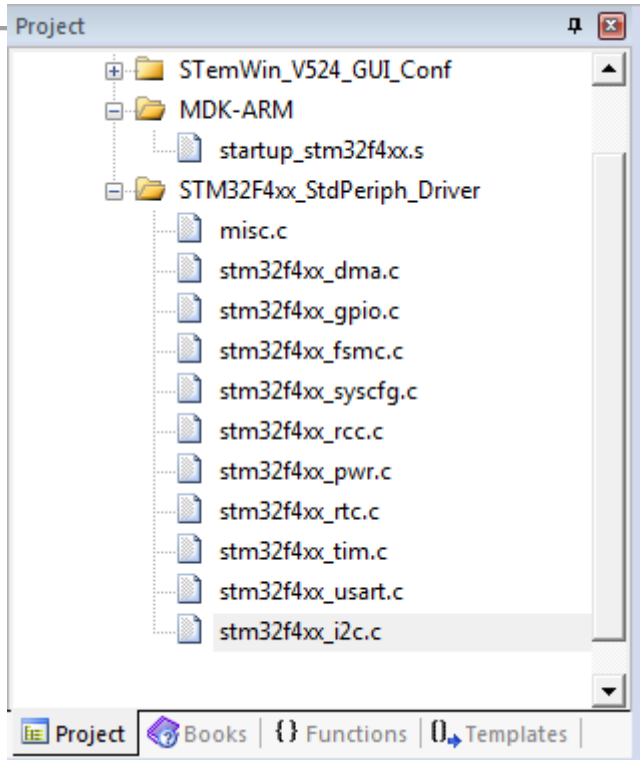
Pro realizaci projetu pro desku STM32F4-Discovery s připojeným LCD dotykovým displejem je nutné nejdříve sestavit základní kostru programu tzn. přidat všechny hlavičkové soubory, knihovny a zdrojové soubory potřebné pro správné fungování desky. Na přiloženém CD nosiči naleznete všechny potřebné soubory, které jsou potřeba pro vytvoření kostry programu, včetně již vytvořené kostry programu pro desku STM32F4-Discovery s připojeným LCD dotykovým displejem. Nyní si ukážeme, jak se kostra programu vytvoří.

Na disku si nejdříve vytvoříme složku, kde budeme ukládat projekt včetně všech souborů, které budeme potřebovat. Do této složky překopírujeme složky Libraries, Utilities ze složky Default Project. Dále zde překopírujeme složky src a inc ze složky Default Project/Project. Ve vývojovém prostředí poté vytvoříme projekt. Vybereme zařízení, které budeme programovat a začneme přidávat potřebné soubory.

Do projektu nejdříve přidáme soubory potřebné pro nastavení LCD displeje, soubory pro nastavení a inicializaci knihovny STemWIN, hlavičkový soubor s globálními proměnnými a knihovnu STemWIN.

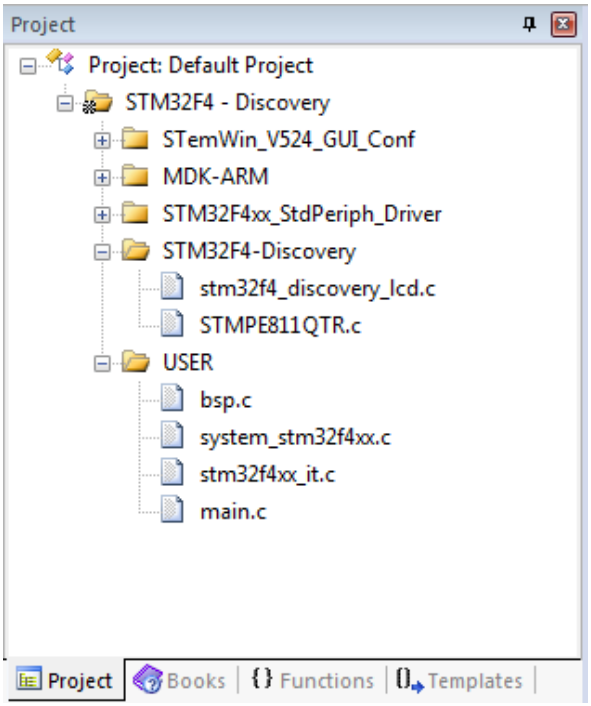


Obr. 45 Soubory pro konfiguraci LCD displeje a knihovny STemWIN

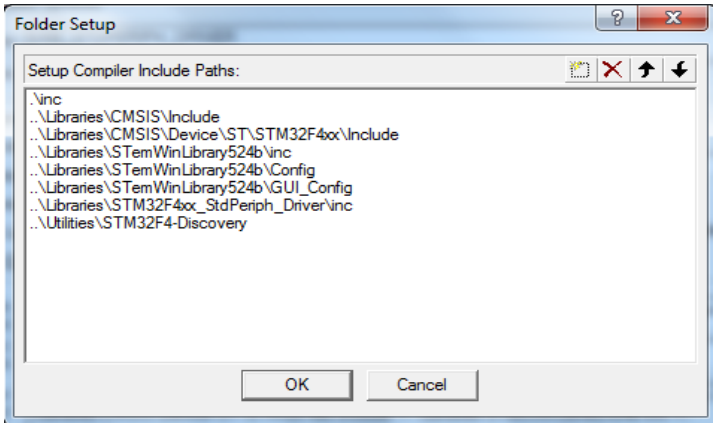


Obr. 46 Drivery pro periférie a soubor pro start

Dále do projektu přidáme soubor potřebný pro start desky a ovladače důležitých periférií, které budou v projektu potřebné. Nakonec je nutné přidat drivery pro LCD displej a zdrojové soubory pro nastavení desky STM32F4 – Discovery. Před kompilací programu je nutné ještě přidat cesty k hlavičkovým souborům pro kompilátor. To provedeme pomocí tlačítka Options for Target v záložce C/C++.V preprocesorových definicích zadáme: USE_STDPERIPH_DRIVER a poté přidáme cesty k hlavičkovým souborům. Pokud nyní zkompilujeme program, nebudou v něm žádné chyby.



Obr. 48 Drivery pro LCD displej a soubory pro nastavení desky STM32F4 - Discovery



Obr. 47 Cesty k hlavičkovým souborům

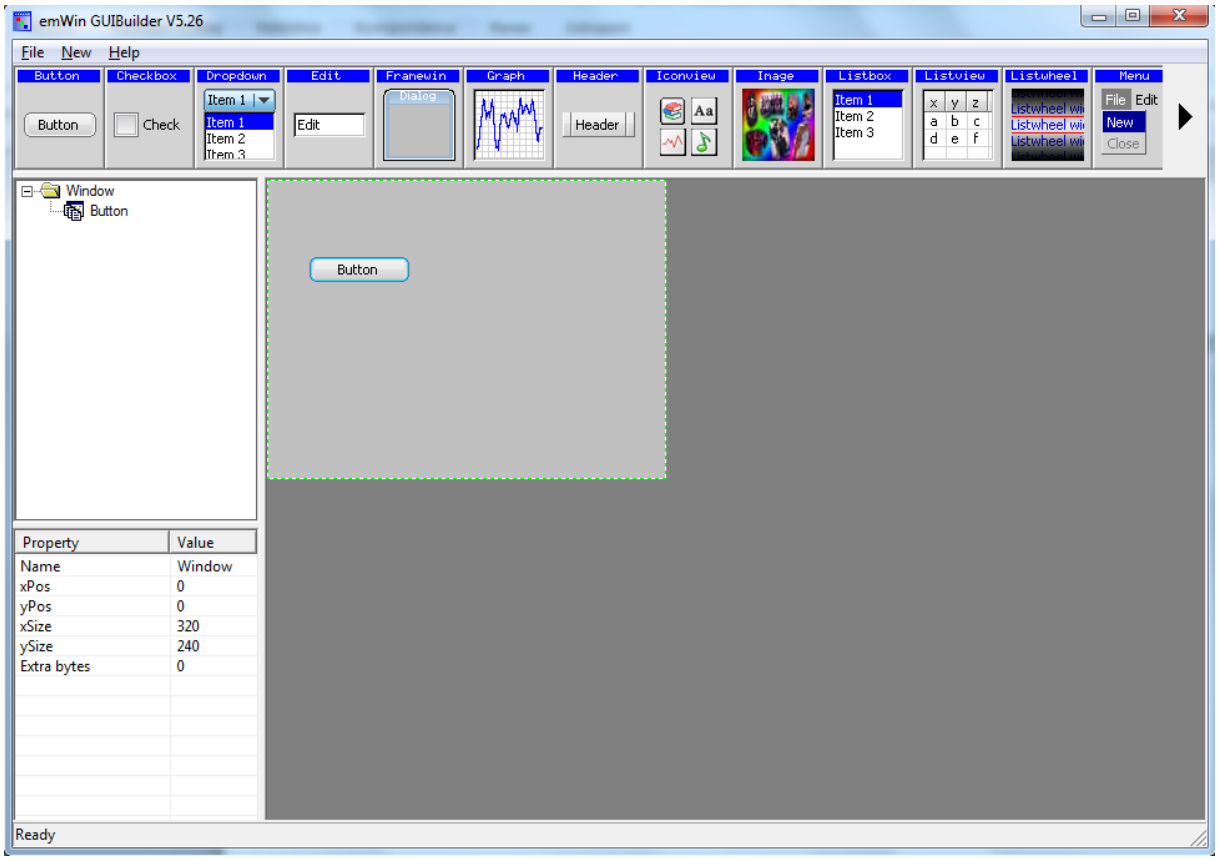
Pro přidání okna je nejjednodušší použít utilitu GUIBuilder. Tato utilita nám dovolí vytvářet okna pomocí grafického rozhraní. Po uložení se nám vygeneruje zdrojový soubor, který můžeme použít v našem projektu. Pro názornost si vytvoříme okno, do kterého umístíme tlačítko. Prvek okna nalezneme na konci lišty s výběrem widgetů, prvek tlačítka se nachází na úplném začátku lišty s widgety.

Po uložení se nám vygeneroval soubor WindowDLG.c, který nalezneme ve složce: Default Project\Libraries\STemWinLibrary524b\Software. Pomocí vývojového prostředí tento soubor přidáme do našeho projektu. Po přidání souboru do projektu musíme ještě do funkce main() přidat tento příkaz pro vytvoření okna, které nám vytvořil GUIBuilder.

```
int main(void){

  BSP_Init();
  WM_SetCreateFlags(WM_CF_MEMDEV);
  GUI_Init();
  hDlgMain=CreateWindow();
  while (1) {
     GUI_Delay(100);
   }
}
```

Obr. 49 Funkce main() po přidání okna



Obr. 50 Utilita GUIBuilder po vytvoření okna a tlačítka

Nyní zkompilujeme program. Kompilátor bude hlásit varování, že funkce CreateWindow() byla deklarována implicitně, ale tohota varování si nemusíme všímat. V nastavení debugeru je ještě

nutné zvolit možnost Download to Flash. Program můžeme nyní nahrát do paměti a po resetování desky by na LCD displeji mělo být vyobrazeno okno s tlačítkem.

B.3 Program pro ovládání LED diody na desce STM32F4- Discovery

Aplikace, která bude ovládat diodu bude pro své fungování vyžadovat 2 tlačítka. První tlačítko bude diodu rozsvěcet a druhé ji bude zhasínat. Toto rozhraní vytvoříme pomocí utility GUIBuilder, která nám návrh značně zjednoduší.

Pro ovládání diody je nutné nejdříve vytvořit funkci, která inicializuje a nastaví pin, ke kterému je dioda připojena. Pro tento účel si vytvoříme funkce LED_Init(), kterou poté zavoláme ve funkci main().

```
void LED_Init(void){

  GPIO_InitTypeDef GPIO_InitStruct;

  RCC_AHB1PeriphClockCmd(RCC_AHB1Periph_GPIOD,ENABLE);

  GPIO_InitStruct.GPIO_Pin = GPIO_Pin_12;
  GPIO_InitStruct.GPIO_Mode = GPIO_Mode_OUT;
  GPIO_InitStruct.GPIO_Speed = GPIO_Speed_50MHz;
  GPIO_InitStruct.GPIO_OType = GPIO_OType_PP;
  GPIO_InitStruct.GPIO_PuPd = GPIO_PuPd_UP;
  GPIO_Init(GPIOD, &GPIO_InitStruct);
}
```

Obr. 51 Definice funkce LED_Init()

Do události po uvolnění tlačítka pro rozsvícení diody jsem poté napsal přikaz: GPIO_SetBits(GPIOD, GPIO_Pin_12). Tento příkaz nastaví na uvedený pin hodnotu 1. Do události po uvolnění tlačítka pro zhasnutí diody jsem napsal příkaz, který nastaví na uvedený pin hodnotu 0.

```
case ID_BUTTON_0: // Notifications sent by 'LED On'
  switch(NCode) {
  case WM_NOTIFICATION_CLICKED:
    // USER START (Optionally insert code for reacting on notification message)
    // USER END
    break;
  case WM_NOTIFICATION_RELEASED:
    // USER START (Optionally insert code for reacting on notification message)


        GPIO_SetBits(GPIOD, GPIO_Pin_12);


    // USER END
    break;
  // USER START (Optionally insert additional code for further notification handling)
  // USER END
  }
  break;
```

Obr. 52 Příkaz pro rozsvícení diody v události po uvolnění tlačítka LED On

```
case ID_BUTTON_1: // Notifications sent by 'LED Off'
 switch(NCode) {
 case WM_NOTIFICATION_CLICKED:
   // USER START (Optionally insert code for reacting on notification message)
   // USER END
   break;
 case WM_NOTIFICATION_RELEASED:
   // USER START (Optionally insert code for reacting on notification message)


       GPIO_ResetBits(GPIOD, GPIO_Pin_12);


   // USER END
   break;
// USER START (Optionally insert additional code for further notification handling)
// USER END
 }
 break;
```

Obr. 53 Příkaz pro zhasnutí diody v události po uvolnění tlačítka LED Off

Obr. 54 Aplikace pro ovládání LED diody na desce STM32F4-Discovery

B.4 Program Calculator

Tento program bude sloužit pro výpočet jednoduchých příkladů ze dvou čísel, které zadá uživatel pomocí grafického rozhraní. Program potřebuje pro svou funkčnost celkem devatenáct tlačítek, kterými uživatel ovládá vstup a výstup. Jedno tlačítko je rezervováno pro návrat programu do menu. Program dále obsahuje dva editační prvky. V prvním editačním prvku je zobrazován vstup, který uživatel zadal a ve druhém je zobrazován průběžný výsledek. Program také obsahuje tlačítko pro s mazání poslední zadané cifry a tlačítko pro smazání celého vstupu a výsledku. Pro usnadnění vytváření rozhraní použijeme opět utilitu GUIBuilder.

Vstup je získán na základě tlačítka, které uživatel stiskne. Hodnota stisknutého tlačítka je poté uložena do pole, jehož index byl získán pomocí funkce strlen(). Obsah pole do něhož je ukládán uživatelský vstup je zobrazováno do editačního pole číslo 1. Po stisknutí tlačítka operace, kterou chce uživatel provést je nejdříve kontrolováno zda nebylo v předcházejícím kroku stisknuto tlačítko pro výpočet resp. tlačítko se znakem '='.

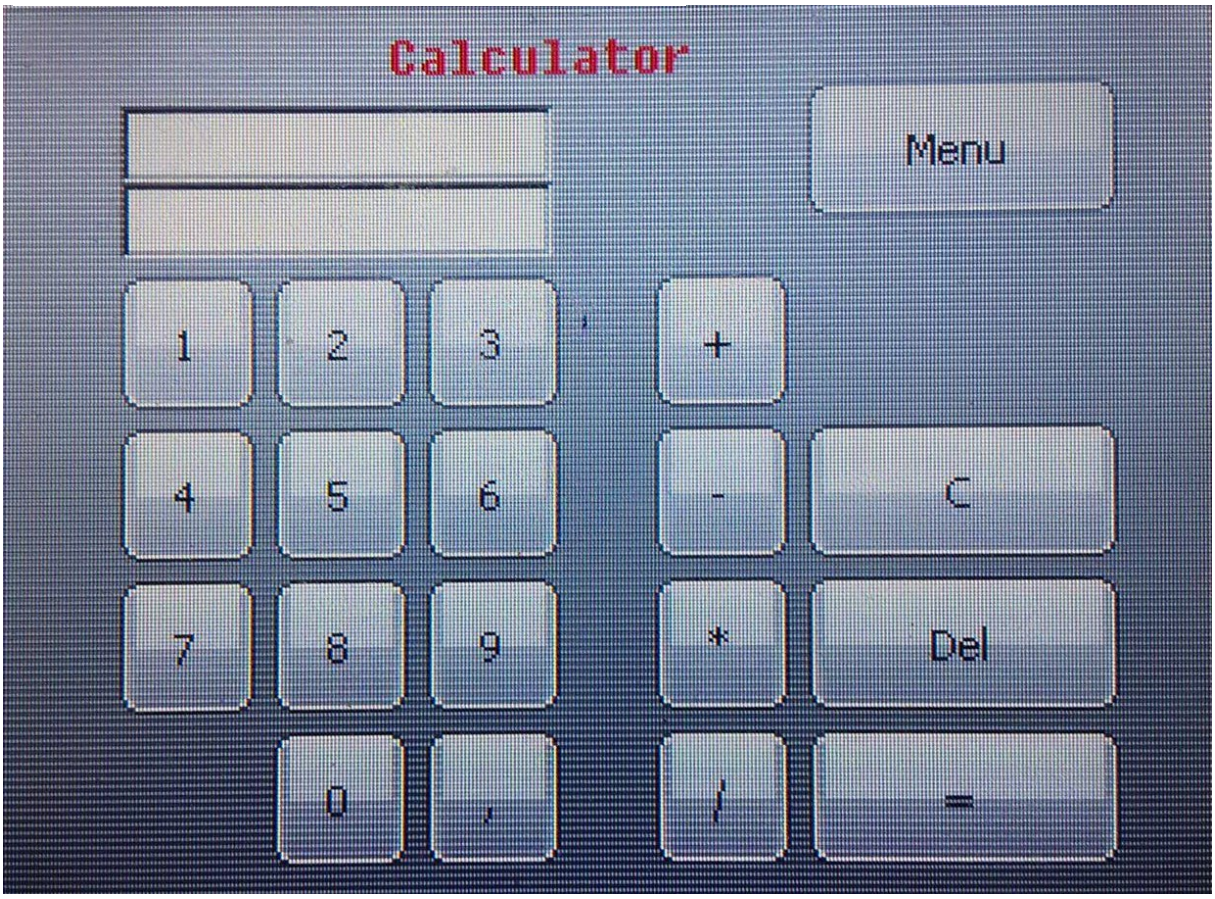
Pokud ano, výsledek operace zůstává stejný. Pokud ne, zkontroluje se zda počet cifer ve vstupní proměnné je menší než 20. V případě, že výsledek tohoto rozhodování je kladný převede se vstup z pole na číslo datového typu double a zavolá se funkce solve(), která provede výpočet na základě operandu, který uživatel stisknul. Výsledek této funkce se pomocí funkce sprintf() převede na pole textových znaků a zapíše se do editačního pole číslo 2. Nakonec se pole vstupu naplní nulami a do editačního pole číslo 1 se tedy nezapíše nic. Celý program naleznete na přiloženém CD.

```
if(opr=='='){
      Vysledek=Vysledek;
      hItem = WM_GetDialogItem(pMsg->hWin, ID_EDIT0);
      EDIT_SetText(hItem, "");
}
else
if ((i=strlen(input))<20)
{
    input[i]='\0';
    sscanf(input, "%lf", &Cislo);
    Vysledek = solve(Vysledek, Cislo, opr);
}
opr = '+';
sprintf(output,"%17.3f",Vysledek);
hItem = WM_GetDialogItem(pMsg->hWin, ID_EDIT1);
EDIT_SetText(hItem, output);
i=strlen(input);
for(n=i;n>=0;n--)
{
  input[n]='\0';
}
```

Obr. 55 Příkazy, které se provedou po stisknutí tlačítka '+'

```
double solve(double V, double C, char O)
{
  switch(O)
  {
    case '+':
      return V + C;
    case '-':
      return V - C;
    case '*':
      return V * C;
    case '/':
      return V / C;
    //case '=':
    default:
      return C;
  }
}
```

Obr. 56 Definice funkce solve()



Obr. 57 Uživatelské rozhraní aplikace Calculator

Shrnutí kapitoly

V této kapitole jsme si stručně popsali podpůrnou knihovnu STM32Cube. Dále jsme si s podporou této knihovny vytvořili celkem 2 příklady. První z nich byla aplikace pro ovládání diody a druhá byla aplikace, která prováděla základní výpočty ze dvou čísel zadaných uživatelem pomocí grafického rozhraní na LCD displeji.

Klíčová slova

STM32Cube, programování, GUIBuilder, ovládání LED, STM32F4 - Discovery

Kontrolní otázka 1

K čemu slouží utilita GUIBuilder?

Tato utilita nám dovolí vytvářet okna, tlačítka, checkboxy a další prvky pomocí grafického rozhraní. Po uložení nám vygeneruje zdrojový soubor s grafikou, kterou jsme si vytvořili.